OpenGIS Project Document 01-044r2

TITLE:	**Units of Measure and Quantity Datatypes**
AUTHOR:	**Name: John Bobbitt**
	Address: POSC **9801 Westheimer Rd., Suite 450** **Houston, TX 77079**
	Phone: 713-267-5174
	Email: bobbitt@posc.org
DATE:	**June 15, 2001**
CATEGORY:	**Discussion Paper**

Status of this Document:

This paper presents a discussion of technology issues considered in a Working Group of the Open GIS Consortium Technical Committee. The content of this paper is presented to create discussion in the geospatial information industry on this topic; the content of this paper is not to be considered an adopted specification of any kind. This paper does not represent the official position of the Open GIS Consortium nor of the OGC Technical Committee.

This OGC Discussion Paper is posted to encourage broad discussion of these draft recommendations. It is posted to help build a consensus among various communities concerned with the encoding of units of measure (UOM) information. Comments on this paper and suggestions for improvement are invited, and can be submitted in email messages to the listed principal author. Please send copies of such email messages to the OGC Coordinate Transformation Working Group, through the email reflector address ct.wg@opengis.org and/or the GML SIG gml.sig@opengis.org. If practical, please send your comments and suggestions before August 13, when a revised version of this document is expected to be prepared.

This version of this Discussion Paper was posted mid-June 2001. This document will evolve over time, and readers should check the OGC web site for later versions in the Fall of 2001, or contact the author for the latest status of this document."

Last revision date: June 12, 2001. The changes to the previous version include:

- Heading block, Contributing authors, references.
- All schema snippets include the xsd prefix.
- Addition of display element to units definitions (in Appendix D, options).

Contributing Authors

The following contributed significantly to the paper and the ideas contained in the paper:

Simon Cox – CSIRO

Don Wilhemlsen – Schlumberger Oil Services

Paul Daisey – U. S. Bureau of Census

Roger Cutler – Chevron Oil Co.

In addition, scores of others contributed through the various email discussion groups. The following paper is a collation of these ideas.

References

XML Schema is accessed at http://www.w3.org/XML/Schema. This area lists tutorials, software, and discussions on XML Schema, as well as having the basic XML Schema Recommendation Papers.

ISO Document Recommendation 20 can be found at http://www.unece.org/cefact/rec/rec20en.htm.

Various Unit of Measure Dictionaries (definitions and symbols) can be found at the following:

Olken and McCarthy, Measurement Units in XML Datatypes: http://pueblo.lbl.gov/~olken/mendel/w3c/xml.schema.wg/units/syntax.htm

POSC; Units of measure - used in POSC data base: http://www.posc.org/Epicentre.2_2/DataModel/LogicalDictionary/StandardValues/ref_unit_of_measure_si_conversion.html

Unified Code for Units of Measure: http://aurora.rg.iupui.edu/~schadow/units/UCUM/

Discussion of the use of the recommendations in this paper: Cox, Simon; Observations in XMML: http://www.ned.dem.csiro.au/XMML/issues/observations.html

The original discussion paper on this topic was developed by John Bobbitt. It can be found at http://www.posc.org/ebiz/pefxml/UnitsOfMeasure.doc

Units of Measure Recommendations

The following are contained in this recommendation report:

1. Recommendation for an abbreviation for units of measure.
2. Recommendation for quantity datatype.
3. Discussion on extensions to more complex types.
4. Recommendation on a Units Dictionary.
5. Recommendation on Units of Measure Definitions.
6. Recommendation on encapsulating Units of Measure Conversions.
7. Recommended pattern of usage of the various classes and datatypes.
8. Documentation of the classes and datatypes.
9. Sample Units of Measure Dictionary.
10. Sample XML file.
11. Options to the above recommendations and patterns.

a) Currencies

b) No skip references

c) Restricted quantity types

d) Alternative formats (such as DMS).

e) Key on symbols instead of URI references.

f) Alternative Units of Measure Conversion Formulas

Usage Needs:

The quantity data type and units definitions classes are based on usage needs as expressed by members of several groups. The recommendations in this paper are given to satisfy these needs to the greatest extent possible. Because of the importance of these needs, they are listed below. The order is not important.

1. Any unit of measure used in a file must be defined in a manner that the reading application may understand it.
2. Any quantity value in an XML file must have a unit of measure that can be determined from the XML file and/or easily referenced files.
3. The method for handling quantity values should be extendable to more complex types of data, such as arrays, tuples, and complex numbers.
4. An application must not be forced to record the unit of measure for each individual value where the unit of measure is the same for a set of values and may be determined from a higher context. The method must allow units of measure to be set within a context.
5. It is best that an XML file should not rely on an outside reference to determine the unit of measure or the meaning of the unit of measure.
6. A writing application must be able to use any appropriate unit of measure. If one is used, it must be able to define the meaning of the unit, or reference where the meaning is defined.
7. A writing application must be able to define a base unit of measure for a particular quantity type. The base unit need not be defined within the context of the SI system of units.
8. The method must allow for cases where a unit of measure is given, but its meaning is unknown. For example, a unit of pressure may be given as pounds per square inch, but it is not known whether this is gauge pressure or absolute pressure.
9. The method should not restrict the unit of measure to a pre-specified list. If an “erroneous” unit of measure is given, it should be up to an application or user intervention to determine the correct unit.
10. The basic units model should match the ISO TC211 units model to the extent possible, realizing that the ISO model is not a general model.
11. The method must balance convenience of use against the proliferation of a great number of different Units of Measure in the recommended schema.
12. A method should be available to allow a standard dictionary to be used, with reference by symbol.
13. The quantities – values and units of measure – must be displayable using XSL and XSLT.

Terminology

A strong recommendation is one which applications can expect to be adopted. Any changes to a strong recommendation must be backward compatible.

A weak recommendation is given where it is expected that some details will change. Applications should consider a weak recommendation to be a best practice that may change in its details in the future. Applications should not consider a weak recommendation to be backward compatible.

Summary of Recommendations

Strong Recommendations:

Recommendation (Strong):

The term, UnitOfMeasure, may be fully spelled out. When abbreviated, uom should be the abbreviation used.

Recommendation (Strong):

The quantity data type be defined as follows (XML Schema):

```
<xsd:complexType name=”quantityType”>
  <xsd:simpleContent>
    <xsd:extension base=”xsd:double”>
      <xsd:attribute name=”uom” type=”xsd:anyURI” use=”optional”/>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>
```

Recommendation (strong): It is strongly recommended that the unitref datatype be used to set a unit of measure that is used in context.

Because of ambiguity among the unit of measure symbols, it is **strongly recommended** that reference to a unit of measure not be made by symbol alone. It is **strongly recommended** that a reference to a unit of measure lead to an external XML file, or a location within the present XML file that defines the unit of measure.

Recommendation (strong, and weak):

It is **strongly recommended** that Units Dictionary XML files be developed for referencing by applications.

It is **weakly recommended** that the complex type, “unitDictionaryEntry,” be used to capture the information. The weak recommendation is given because there has not been enough usage of the class to insure that it is complete and implementable. Further development of the unitDictionaryEntry complex type should occur during the next few months.

Recommendation(strong):

It is **strongly recommended** that the conversionType data type be used to give conversions to a base unit. It is not anticipated that this class will change.

Weak Recommendations:

Recommendation (Weak):

For arrays and tuples of values, the units be incorporated within the main tag.

There is no recommendation about how the array or tuples of values are to be represented.

Recommendation (weak):

It is **weakly recommended** that the units definition block of an XML file use the unitDefintionType datatype. The weak recommendation is given because there has not been enough usage of the Class to insure that it is complete and implementable. Further development of the unitDefinitionType complex type should occur during the next few months.

Recommendation (weak):

It is **weakly recommended** that the Base Pattern be used for using and defining the units of measure..

Quantity Data Types

A quantity is a value, or a list of values, and a unit of measure. A **strong recommendation** is given for this datatype for a single value.

In addition, **weak recommendations** are given for how to extend this to more complex datatypes than a single value. These complex datatypes include arrays and tuples. We need more guidance on encoding and working with arrays, tuples, structures, and other complex datatypes. When these methods become clearer, strong recommendations will be given for incorporating the units of measure into these more complex structures.

Abbreviation (Strong Recommendation)

Discussion: The full term, Units of Measure, is often abbreviated in XML elements. There are two obvious abbreviations that are both widely used: unit and uom. Initially it appears that the appropriate abbreviation is simply a matter of choice. However, because the term, unit, can have semantic meaning other than an abbreviation for Unit of Measure, it is less ambiguous to use the abbreviation uom. (Examples of such meaning are: Prudhoe Bay Unit, Military Unit, Cardiology Unit.) The term, uom, has no semantic meaning other than as an abbreviation for Units of Measure.

Recommendation (Strong):

The term, UnitOfMeasure, may be fully spelled out. When abbreviated, uom should be the abbreviation used.

Examples:

Examples of tags using the abbreviation would be

uom

uomReference

UomDictionary

Quantity Data Type (Strong Recommendation)

Discussion: A quantity data type is a value plus a unit of measure. The unit of measure is a reference to a definition of the unit of measure. The reference may be a well-known symbol, or a URI Reference to an XML definition of the unit. The URI Reference has been chosen as appropriate, since it allows more flexibility (see the Patterns of Usage section).

The value may be associated with the unit in one of two ways: *within a tag*, and *within a block*. These two methods appear as below:

```
  Within a tag (XML):
```

```
<lengthOfSide uom=”#ft”>84.6</lengthOfSide>

Within a block (XML):
<lengthOfSide>
  <uomReference>#ft</uomReference>
  <amount>84.6</amount>
</lengthOfSide>
```

There appears to be a strong preference for the *within a tag* approach.

In addition, the *Within a Tag* approach appears to marginally better for these reasons: firstly, it is more compact, although that alone would not recommend it. More importantly, it is expected that the Quantity datatypes will be incorporated into various referenced schema, and that the child tags will inherit the namespace properties of the referenced schema. There would be namespace issues with the *Within a Block* approach, whereas the *Within a Tag* approach does not have such issues.

Recommendation (Strong):

The quantity data type be defined as follows (XML Schema):

```
<xsd:complexType name=”quantityType”>
  <xsd:simpleContent>
    <xsd:extension base=”xsd:double”>
      <xsd:attribute name=”uom” type=”xsd:anyURI” use=”optional”/>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>
```

Incorporation into Schema: For every element that has its uom included with it, the application schema should use the quantityType as follows: (3 Examples)

```
<xsd:element name=”lengthOfSide” type=”quantityType”/>
<xsd:element name=”bottomholePressure” type=”quantityType”/>
<xsd:element name=”gasOilRatio” type=”quantityType”/>
```

Comments and Issues:

Issue: Should the uom attribute be required? A need for the following rule has been expressed. If the unit of measure is missing, the unit should be inherited from a higher level within the context. The uom attribute is given as optional.. With the attribute optional, it is necessary to define the behavior intended if the uom attribute is missing.

1. If the uom attribute is present, the unit of measure assigned to the quantity value is the one referenced by the uom.
2. If the uom attribute is missing:
   a) Obtain the uom last set in context. I.e., travel upwards through the XPATH until you encounter the setting of a uom.
   b) The uom in the context must be set using a uom attribute at a higher level.
   c) Only one uom can be active in a context.
3 The most general uom reader should also test for a third possibility. It may be that the quantityType was altered to set a default unit of measure. The most general reader should check the schema to see if such a default has been set.

The proposed quantityType requires that 1 and 2 be considered. If the quantityType is restricted so that the uom attribute is required, a reader only need consider 1. If the quantityType is altered so that a default value is set, 1, 2, and 3 must be considerd.

Issue: The anyURI datatype requires that an instance be present at the referenced URI. This does not allow reference by well-know symbol. However, the patterns of usage below show that this type of reference is possible. (See Appendix D).

Note: The examples all show references with "unit symbols" as the final reference. This is a convenience for readability. In particular, the URI reference keys to an instance with an ID data type which matches the URI reference value. There is no requirements being placed on the values of this attribute, other than the W3C requirement inherent in the ID datatype.

Units of Measure within a context (Strong Recommendation)

Discussion: There are times when a unit of measure should be valid within a context. Examples (XML):

- Setting the unit of measure for a coordinate system axis.

```
<CoordinateSystemAxis>
  ...
  <uomReference uom="#m"/>
  ...
</CoordinateSystemAxis>
```

- Setting the unit of measure for a set of properties. The following is an example of how the context may be done, and is not being recommended as a best practice.

```
<DimensionsOfBuilding>
  <linearValues uom="#ft"/>
    <length>122</length>
    <width>94.3</width>
  </linearValues>
    ...
  <areaValues uom="#sqft"/>
    <firstFloorArea>10102</firstFloorArea>
    <secondFloorArea>9662</secondFloorArea>
  </areaValues>
    ...
</DimensionsOfBuilding>
```

These may be set using or extending the unitref datatype. Consistent with the quantity data type, this datatype sets the unit within a tag by reference to a URI.

```
<xsd:complexType name="unitref">
  <xsd:attribute name="uom" type="xsd:anyURI" use="required"/>
</xsd:complexType>
```

Recommendation (strong): It is strongly recommended that the unitref datatype be used to set a unit of measure that is used in context.

Note: In the example given, the various tags are not quantity datatypes. They are real numbers, probably of type xsd:double or xsd:decimal. The application schema must understand when it is setting the units within a context, so that it should not use the quantity datatype for the quantity values.

Note: In the first example, the semantics of the unit of measure is that values given for the axis being instantiated are in "#m" (metres). This information must be given in the textual description of the <CoordinateSystemAxis> definition so that an application can be developed that understands its meaning.

It is generally considered bad practice to set a unit of measure context for the whole document, unless there is a very limited set of data

Extensions (Weak Recommendations):

List of Values

Discussion: A quantity data type is a single value with a unit of measure. There are many cases where there is a list of values (1D array), all of which have the same unit of measure. This recommendation does not strongly define how this should be done. However, it suggests the method by which the List Of Values uom should be incorporated into a schema.

Following are two examples of XML for a list of values. The important part is the method by which the uom is incorporated.

```
Examples:
<distanceFromStart uom="#ft">78.2 238.1 344.0 511.2
  </distanceFromStart>

<distanceFromStart uom="#ft">
  <amount>78.2</amount>
  <amount>238.1</amount>
  <amount>344.0</amount>
  <amount>511.2</amount>
</distanceFromStart>
```

Recommendation (Weak):

The units be incorporated within the main tag.

There is no recommendation about how the list of values is to be represented.

Array of Values:

Discussion: An array of values is similar to the list of values (a 1D array), in that every value in the array has the same unit of measure. The particular details of how the elements of the array are expressed is not a part of this recommendation.

Recommendation (Weak):

The units be incorporated within the main tag.

There is no recommendation about how the array of values is to be represented.

Tuples of Values:

Discussion: A tuple is a vector of values. The vector will have one or more components, each of which may have a different unit of measure. An example of a 3-tuple would be a ground location with an x-value, a y-value, and a z-value as the three components.

In general, a tuple of values can be an array of tuples. The array may be a single vector, a list of vectors, or an array of vectors. Each of these cases is an extension of (respectively) the quantity data type, the list of values, and the array of values.

A tuple will require a (standardized) construction to express the values. The assignment of the unit of measure to each component should / must follow the (rules of the standardized) construction. One way this may be done is shown below (for a list of tuples, which here is used to define a wellbore path):

```
Example:
<lineString SRSRef="#localSystem">
  <xValues uom="#m">18.2 97.4 122.0</xValues>
  <yValues uom="#m">-6.3 0.1 –9.22</yValues>
  <zValues uom="#ft">98.3 –257.2 –599.1</zValues>
</lineString>
```

Recommendation (Weak):

The units of measure be incorporated within each component tag in a manner consistent with the structure of the data values.

There is no recommendation about how the tuple values should be represented. Specifically, the example XML fragment should not be construed as the recommended standardized construction.

Units Definition

Once a quantity value has a unit of measure assigned to it, it is necessary to understand the meaning of that unit of measure; and, optionally, how to display the unit. In general, the meaning of a unit of measure is its conversion to a base unit (or else it is a base unit).

From the user requirements, it is necessary that the reading application understand the unit of measure. It may do so by having a pre-existing knowledge of the unit, or by determining the meaning of the unit from the given reference to it.

Because of ambiguity among the unit of measure symbols, it is **strongly recommended** that reference to a unit of measure not be made by symbol alone. It is **strongly recommended** that a reference to a unit of measure lead to an external XML file, or a location within the present XML file that defines the unit of measure.

Units Definitions will be given in two classes. One is appropriate for a Units Dictionary, and the other is designed for an inline definition. However, both are still in flux.

They are designed to give key information that will allow a reading application to understand the unit of measure. They are also designed to give additional information that will help a human reader understand the unit of measure, but the information is not critical to tying the unit of measure to its base unit.

The key information is whether the unit of measure is a base unit or not. If it is not a base unit, the key information is the conversion to a base unit. It is expected that these components will not change in later versions of these classes.

The other information may change as users determine that the information captured is incomplete. However, the key information for a unit of measure service should not change.

Units Dictionary

Discussion: A quantity type references a unit of measure. It is recommended that the definition of the unit be available in one or more unit dictionaries. The dictionaries are to be XML files based on the schema defined below. They may be separate files accessible across the internet, or the dictionary may be inline in the XML file being read.

XML Schema: Following is an incomplete schema for a Units Dictionary file. The complex data types, “conversion,” “baseUnitType,” and “explicitType” (namespace ‘my’) are given in Appendix A. The “docInfoType” (namespace ‘localsite’) is not specified. It is expected that the DocumentInformation block will be site specific, and is not a part of this recommendation.

```
<!--Usage of the type-->

<xsd:element name=”UnitOfMeasureDictionary”>
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element name=”DocumentInformation”
                type=”localsite:docInfoType”/>
   <xsd:element name=”UnitsDefinition”>
    <xsd:complexType>
```

```
       <xsd:sequence>
        <xsd:element name="UnitOfMeasure" type="my:unitDictionaryEntry"
           minOccurs="1" maxOccurs="unbounded"/>
       </xsd:sequence>
      </xsd:complexType>
     </xsd:element>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>

  <!--The type defined. See Option, display element, in the appendix-->

  <xsd:complexType name="unitDictionaryEntry">
    <xsd:sequence>
      <xsd:element name="name" type="xsd:string" minOccurs="0"/>
      <xsd:element name="measurementType" type="xsd:string"
                   minOccurs="0"/>
      <xsd:element name="catalogName" type="xsd:string"/>
      <xsd:element name="catalogSymbol" type="my:explicitType"/>
      <xsd:choice>
        <xsd:element name="BaseUnit" type="my:baseUnitType"/>
        <xsd:element name="ConversionToBaseUnit" type="my:conversion"
             minOccurs="0"/>
      </xsd:choice>
    </xsd:sequence>
    <xsd:attribute name="uid" type="xsd:ID" use="required"/>
    <xsd:attribute name="annotation" type="xsd:string" use="optional"/>
  </xsd:complexType>
```

XML Example:

See Appendix B for an example of a Units Dictionary.

Recommendation (strong, and weak):

It is **strongly recommended** that Units Dictionary XML files be developed for referencing by applications.

It is **weakly recommended** that the complex type, "unitDictionaryEntry," be used to capture the information. The weak recommendation is given because there has not been enough usage of the class to insure that it is complete and implementable. Further development of the unitDictionaryEntry complex type should occur during the next few months.

Appendix A gives the full schema for the unitDictionaryEntry along with an explanation of the elements.

The section of the schema above headed ***"<!--Usage of the Type-->***" and *italicized,* is not part of the recommendation. See Patterns of Usage later for more details on usage.

Units Block

Discussion: Within a file, it is recommended that a block be set aside for the units definitions. All uses of units of measure with quantities should be given through anyURI references. The (weakly) recommended pattern is that all uses in quantity instances will referenced internal definition instances. The unit block will contain the ID's for the internal references.

The units block must have one of two types of entries. The first choice is a "skip[1]" reference, generally to a units dictionary as described above. The other choice is an inline definition of the unit of measure itself. The former allows an application domain to control the set of units of measure through specifying a dictionary (or dictionaries) to be used. The latter (inline definition) choice allows a file to define units of measure that are not in any referenced dictionary.

The schema fragment for the complex Datatype, "unitDefinitionType" is given below. It has one additional (optional) element that the unitDictionaryEntry above does not have: unknown. This is based on a user requirement, that a unit of measure symbol is known, but its meaning (i.e., its conversion factor) often is not known. This is often the case with legacy data. For example, a legacy file may show that the units are FT (feet), but it is not known which type of foot is being used (US statutory foot, International foot, etc.).

XML Schema:

The XML Schema for this is similar to the one for unitDictionaryEntry above. It is given below, again, and some of the data types are included in Appendix A.

```
<!-Usage of the Type-->

<xsd:element name="UnitOfMeasureBlock"
                     type="my:unitDefinitionEntry"/>

<!--The types defined-->

<xsd:complexType name="uomBlockEntry">
  <xsd:choice maxOccurs="unbounded">
    <xsd:element name="uomReference" type="my:refToType"/>
    <xsd:element name="UnitOfMeasure" type="my:unitDefinitionType"/>
  </xsd:choice>
</xsd:complexType>

<xsd:complexType name="unitDefinitionType">
  <xsd:sequence>
    <xsd:element name="name" type="xsd:string" minOccurs="0"/>
    <xsd:element name="measurementType" type="xsd:string"
              minOccurs="0"/>
    <xsd:element name="catalogName" type="xsd:string" minOccurs="0"/>
    <xsd:element name="catalogSymbol" type="my:explicitType"
          minOccurs="0"/>
    <xsd:choice>
      <xsd:element name="BaseUnit" type="my:baseUnitType"/>
      <xsd:sequence minOccurs="1">
        <xsd:element name="unknown" minOccurs="0">
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="ConversionToBaseUnit" type="my:conversion"
           minOccurs="0"/>
      </xsd:sequence>
    </xsd:choice>
  </xsd:sequence>
  <xsd:attribute name="uid" type="xsd:ID" use="required"/>
  <xsd:attribute name="annotation" type="xsd:string" use="optional"/>
</xsd:complexType>
```

[1] A skip reference is taken from the metaphor of a skipping rock. The rock hits the water, and skips to another place. A skip reference performs the same function. It takes a reference to itself (based on its ID value, and invokes a reference to another location (based on an anyURI value).

XML Example:

An example of a file that includes a units definition block is given in Appendix C.

Recommendation (weak):

It is **weakly recommended** that the units definition block of an XML file use the above datatype(s). The weak recommendation is given because there has not been enough usage of the Class to insure that it is complete and implementable. Further development of the unitDefinitionType complex type should occur during the next few months.

Appendix A gives the full schema for the unitDictionaryEntry along with an explanation of the elements.

Note that the section of the schema above headed ***"<!--Usage of the Type-->*"** and *italicized,* is not part of the recommendation. See Patterns of Usage later for more details on usage.

Units Conversion

Discussion: The units conversion information is carried by the complex datatype, "conversion." It encapsulates the general formula, Y = (A + BX) / (C + DX) where X is the value of the unit to be converted, and Y is the value of the unit converted to (generally the base unit). A, B, C, and D are the parameters of the conversion. They are given in the conversion datatype with element names of firstTerm, secondTerm, thirdTerm, and fourthTerm.

Simplifications are as follows. If A = D = 0, and C = 1, the remaining term, B, becomes a "conversion factor. For example, the conversion from international foot to metre has a factor = .3048. If A = D = 0, the conversion factor becomes a numerator / denominator (B / C). For example, the conversion from US statutory foot to metre has a numerator = 12., and a denominator = 39.37. These two simplifications are captured in alternative ways to give the conversion.

The full formula is needed for some conversions. For example, the conversion from Kelvin to Celsius or Fahrenheit degrees, the conversion from gauge pressure to absolute pressure, and the conversion from API gravity to density all require either A or D to be non-zero.

See Appendix D, Option: Alternative Units of Measure Conversion Formulas, for more detail on the units conversion model.

XML Schema:

The XML Schema for the complex type, "conversion" is given below. It is explained more fully in Appendix A.

```
<xsd:complexType name="conversion">
  <xsd:sequence>
    <xsd:choice>
      <xsd:element name="factor" type="xsd:double"/>
      <xsd:sequence>
        <xsd:element name="numerator" type="xsd:double"/>
        <element name="denominator" type="xsd:double"/>
      </xsd:sequence>
      <xsd:sequence>
        <xsd:element name="firstTerm" type="xsd:double"
            minOccurs="0"/>
        <xsd:element name="secondTerm" type="xsd:double"/>
        <xsd:element name="thirdTerm" type="xsd:double"/>
        <xsd:element name="fourthTerm" type="xsd:double"
            minOccurs="0"/>
      </xsd:sequence>
    </xsd:choice>
    <xsd:element name="description" type="xsd:string" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="baseUnit" type="xsd:anyURI" use="required"/>
</xsd:complexType>
```

XML Example:

Sample XML files are given in both Appendixes B and C.

Recommendation(strong):

It is **strongly recommended** that the conversion data type be used to give conversions to a base unit. It is not anticipated that this class will change.

Patterns of Usage

Patterns of Usage have been implicit in the above topics. Consistent patterns of the data are as important as the element tags themselves. Consistent patterns allow applications to encounter the information in a predictable fashion.

When an application schema is developed, the patterns are given. However, this document is not specifying an application schema. Rather, it is specifying Datatypes and Classes, which are to be gathered together into Applications. In doing so, it is useful to set guidelines on how these Datatypes and Classes should be pieced together to give a usable and predictable pattern.

The recommendations given below are not strong. They are given as guidelines. The strong specifications will come when application schema are produced. The extent to which they conform to similar patterns of usage will determine their interoperability.

Base Pattern

The base pattern is the suggested pattern to use. It will be described. In Appendix D, optional patterns of handling units of measure will be described. . All patterns should be described assuming a Quantity Datatype where the unit is used, and a Units Definition Class where the unit is defined.

Use of Units of Measure

All data to be transferred should be analyzed to determine if the data values should include a unit of measure. Where appropriate, those elements should be of type quantityType, or similar types for arrays and tuples. Within the XML file itself, the uom attribute should reference a locally defined unit of measure.

Example:

```
  <!--XSD (XML Schema) file-->
  <xsd:element name=”lengthOfSide” type=”my:quantityType”/>

  <!--XML file-->
    <lengthOfSide uom=”#ft”>103.7</lengthOfSide>
```

Note that the local reference requires the existence of an instance with ID=”ft” to exist within the file. Since all references are local, this usage will avoid possible conflicts with global meanings of the particular ID value used.

Defining the Units of Measure

For every unit of measure used in the XML file, there must be an instance defining the unit. This should be done in a single “Units Block” section as defined earlier in this document. Typically, the Units Block section will be near the beginning or ending of a file, but it may appear anywhere within the file. In some application scenarios, a company or group may define a Units Block section that will be “pasted” into any XML file in order to maintain consistent usage.

The purpose of the Units Block section is to either reassign the local reference to a global reference (the skip reference), or to define the unit of measure inline. Here is an example file that shows both. Note that the skip references may reference multiple units dictionaries.

XML Example:

```
<UnitOfMeasureBlock>
  <uomReference
     uid="m"
     To="http://www.posc.org/applications/unitsDict.xml#m"/>
  <uomReference
     uid="ft1"
     To="http://www.posc.org/applications/unitsDict.xml#ft"/>
  <uomReference
     uid="ft2"
     To="http://www.posc.org/applications/unitsDict.xml#ftUS"/>
  <uomReference
     uid="acre"
     To="http://www.goober.com/unitsDictionary.xml#acre"/>
  <UnitOfMeasure
     uid="vara" annotation="vara">
    <name>Texas vara (modern)</name>
    <ConversionToBaseUnit baseUnit="#m">
      <factor>.8466</factor>
    </ConversionToBaseUnit>
  </UnitOfMeasure>
</UnitOfMeasureBlock>
```

Units Dictionary

A Units Dictionary is a method for controlling the meaning of the units of measure. Although the units can be defined each time within an XML file, it is useful to have an outside source of definition of the units, and to have the XML file reference this.

This method has several advantages. One of the most important is that this pattern is useful for any type of referencing – not just units of measure. For example, a coordinate reference system dictionary can be available that has definitions for coordinate systems, transformations, etc. A second advantage is that such a dictionary can be maintained as the primary definition of units of measure approved by a particular body (or, of their coordinate systems, or transforms, etc). Also, it allows the dictionary to be a (software) service[2], rather than an XML file, that can handle the units of measure issues. That is, the skip reference to a dictionary can be captured by an application without actual reference to a dictionary either as a local (on-disk) file or as a URL (remote file).

Appendix B has an example of a Units Dictionary.

Appendix A: Documentation of Units Types

This appendix contains a full schema of the data types and classes introduced in the above document. It also contains a description of the elements.

The full XML Schema will be given. The sections following will then consider the schema for each datatype defined.

[2] By a (software) service, we envision an application that will take as input the anyURI value, and return an appropriate piece of information. This document does not intend to specify any details about such a service – only the fact that one can exist and can be referenced this way.

The various complexTypes are given either as a complex datatype or a class. Within the meaning of XML Schema, there is no difference in the two concepts. In this document, there is no normative difference intended in the two concepts. However, it is useful to think of a class as being a more complex structure that is intended to capture a set of information, while a complex datatype captures a single piece (although a complex piece) of information.

Note: The schema defines a target namespace. This setting is not a part of the recommendation. There are no recommendations being given for namespace strategies.

Full XML Schema:

```
<?xml version="1.0"?>

<xsd:schema
     targetNamespace="http://www.posc.org/schemas"
     xmlns="http://www.posc.org/schemas"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

<xsd:complexType name=”quantityType”>
 <annotation>
   <documentation>The XML Schema primitive data type “double” is used
because it is a single representation that offers the greatest
flexibility and precision. However, there is no requirement to store
the data using any particular machine representation, and
applications receiving tdata in elements of this type may choose to
coerce the data to any other type as convenient.</documentation>
 </annotation>
  <xsd:simpleContent>
    <xsd:extension base=”xsd:double”>
      <xsd:attribute name=”uom” type=”xsd:anyURI” use=”optional”/>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>

<xsd:complexType name="refToType">
  <xsd:attribute name="uid" type="xsd:ID" use="required"/>
  <xsd:attribute name="To" type="xsd:anyURI" use="required"/>
</xsd:complexType>

<xsd:complexType name="unitref">
  <xsd:attribute name="uom" type="xsd:anyURI" use="required"/>
</xsd:complexType>

<xsd:complexType name="uomBlockEntry">
  <xsd:choice maxOccurs="unbounded">
    <xsd:element name="uomReference" type="refToType"/>
    <xsd:element name="UnitOfMeasure" type="unitDefinitionType"/>
  </xsd:choice>
</xsd:complexType>

<xsd:complexType name="unitDefinitionType">
  <xsd:sequence>
    <xsd:element name="name" type="xsd:string" minOccurs="0"/>
    <xsd:element name="measurementType" type="xsd:string"
         minOccurs="0"/>
    <xsd:element name="catalogName" type="xsd:string" minOccurs="0"/>
    <xsd:element name="catalogSymbol" type="explicitType"
```

```
         minOccurs="0"/>
    <xsd:choice>
      <xsd:element name="BaseUnit" type="baseUnitType"/>
      <xsd:sequence minOccurs="1">
        <xsd:element name="unknown" minOccurs="0">
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="ConversionToBaseUnit"
            type="conversionType" minOccurs="0"/>
      </xsd:sequence>
    </xsd:choice>
  </xsd:sequence>
  <xsd:attribute name="uid" type="xsd:ID" use="required"/>
  <xsd:attribute name="annotation" type="xsd:string" use="optional"/>
</xsd:complexType>

<xsd:complexType name="unitDictionaryEntry">
  <xsd:sequence>
    <xsd:element name="name" type="xsd:string"/>
    <xsd:element name="measurementType" type="xsd:string"
      minOccurs="0"/>
    <xsd:element name="catalogName" type="xsd:string" minOccurs="0"/>
    <xsd:element name="catalogSymbol" type="explicitType"
          minOccurs="0"/>
    <xsd:choice>
      <xsd:element name="BaseUnit" type="baseUnitType"/>
      <xsd:element name="ConversionToBaseUnit"
               type="conversionType"/>
    </xsd:choice>
  </xsd:sequence>
  <xsd:attribute name="uid" type="xsd:ID" use="required"/>
  <xsd:attribute name="annotation" type="xsd:string" use="optional"/>
</xsd:complexType>

<xsd:complexType name="explicitType">
  <xsd:simpleContent>
    <xsd:extension base="xsd:string">
      <xsd:attribute name="explicit" use="optional">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="Yes"/>
            <xsd:enumeration value="No"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>

<xsd:complexType name="baseUnitType">
 <xsd:sequence>
  <xsd:element name="description" type="xsd:string" minOccurs="0"/>
  <xsd:element name="basicAuthority" type="xsd:string"
         minOccurs="0"/>
 </xsd:sequence>
</xsd:complexType>
```

```
<xsd:complexType name="conversionType">
  <xsd:sequence>
    <xsd:choice>
      <xsd:element name="factor" type="xsd:double"/>
      <xsd:sequence>
        <xsd:element name="numerator" type="xsd:double"/>
        <xsd:element name="denominator" type="xsd:double"/>
      </xsd:sequence>
      <xsd:sequence>
        <xsd:element name="firstTerm" type="xsd:double"
          minOccurs="0"/>
        <xsd:element name="secondTerm" type="xsd:double"/>
        <xsd:element name="thirdTerm" type="xsd:double"/>
        <xsd:element name="fourthTerm" type="xsd:double"
          minOccurs="0"/>
      </xsd:sequence>
    </xsd:choice>
    <xsd:element name="description" type="xsd:string" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="baseUnit" type="xsd:anyURI" use="required"/>
</xsd:complexType>

</xsd:schema>
```

Descriptions of the Datatypes and Classes

Complex Datatype: quantityType

```
<xsd:complexType name=”quantityType”>
  <xsd:simpleContent>
    <xsd:extension base=”xsd:double”>
      <xsd:attribute name=”uom” type=”xsd:anyURI” use=”optional”/>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>
```

An element declared to be a quantityType datatype will have a value and an URI reference to a unit of measure.

element value: The numerical value of the quantity element. ***Note:*** The numerical value is of type double. This is not intended to imply anything about the storage of the data on a computer. Double was chosen because it allows a numerical value to be presented in ASCII characters in a decimal format (876.5432) or in an exponential format (8.765432E2).

attribute - uom: A URI reference to an element which defines the unit of measure of the quantity.

Complex Datatype: refToType

```
<xsd:complexType name="refToType">
  <xsd:attribute name="uid" type="xsd:ID" use="required"/>
  <xsd:attribute name="To" type="xsd:anyURI" use="required"/>
</xsd:complexType>
```

An element of type refToType will be an empty element which contains two attributes. The first is of type ID and is used to reference this element. The second is a URI reference to another element. The refToType datatype is used to construct the skip reference elements referred to in this document.

attribute – uid: A unique (within the file) value for referencing this element.

attribute – To: A URI reference to an element which defines the unit of measure for this skip reference.

Complex Datatype: unitref

```
<xsd:complexType name="unitref">
  <xsd:attribute name="uom" type="xsd:anyURI" use="required"/>
</xsd:complexType>
```

An element of type unitref is empty content, and contains a single attribute which references a unit of measure. It is intended as a datatype that can be used to set a unit of measure for a particular context.

attribute – uom: A URI reference to an element which defines the unit of measure of the context.

Complex Datatype: uomBlockEntry

```
<xsd:complexType name="uomBlockEntry">
  <xsd:choice maxOccurs="unbounded">
    <xsd:element name="uomReference" type="refToType"/>
    <xsd:element name="UnitOfMeasure" type="unitDefinitionType"/>
  </xsd:choice>
</xsd:complexType>
```

An element of type uomBlockEntry is intended to conatin a block defining all of the units of measure referenced in the XML file. The block will contain one entry for each unit of measure. The entry will either be a skip reference or a definition of the unit of measure.

element – uomReference: The uomReference element is a skip reference. It is of type refToType.

element – UnitOfMeasure: A block of information containing the definition of the unit of measure. The UnitOfMeasure element is of type unitDefinitionType.

Class: unitDefinitionType

Note: See Options, display element, in appendix D for an additional element.

```
<xsd:complexType name="unitDefinitionType">
  <xsd:sequence>
    <xsd:element name="name" type="xsd:string" minOccurs="0"/>
    <xsd:element name="measurementType" type="xsd:string"
                 minOccurs="0"/>
    <xsd:element name="catalogName" type="xsd:string" minOccurs="0"/>
    <xsd:element name="catalogSymbol" type="explicitType"
                 minOccurs="0"/>
    <xsd:choice>
      <xsd:element name="BaseUnit" type="baseUnitType"/>
      <xsd:sequence minOccurs="1">
        <xsd:element name="unknown" minOccurs="0">
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="ConversionToBaseUnit"
          type="conversionType" minOccurs="0"/>
      </xsd:sequence>
    </xsd:choice>
  </xsd:sequence>
  <xsd:attribute name="uid" type="xsd:ID" use="required"/>
  <xsd:attribute name="annotation" type="xsd:string" use="optional"/>
</xsd:complexType>
```

An element of type unitDefinitionType is a sequence of elements, many of which are optional, whose purpose is to give a definition of a unit of measure. The subelements include references to catalogs and catalog symbols so that global meaning can be given to the units of measure. The element also includes a flag (unknown) to alert the user that the exact meaning of the unit of measure is unknown.

element – name: A name of the unit of measure. This is an uncontrolled string. In general, there is little use that an application can make of the unit name. This value is optional.

element – measurementType: The measurement type that the unit is used for. This value is an uncontrolled string. It is included for informative purposes. It is expected that the application will not make use of this value. Examples of measurement type would be length, time, molar concentration, and porosity. The value is optional.

element – catalogName: For global understanding of a unit of measure, it is often possible to reference a catalog and a symbol in that catalog. This is an uncontrolled string value, and serves as a namespace for the catalog symbol value. This value is optional.

element – catalogSymbol: The catalogSymbol is of type explicitType. It contains a value and an attribute. The value is a symbol that is considered to be unique within the context of the catalogName namespace. The attribute is a flag (Yes or No) which indicated if the symbol appears explicitly in the catalog, or if it is an implicit combination of symbols combined with an algebra of units. For example, the symbol "cm" may be implicit, which might indicate that it is the "m" symbol combined with the "c" prefix. This element is optional.

choice: There is a mandatory choice.

> **element BaseUnit:** All units of measure which can be converted to each other form an equivalence class of units. The base unit is the representative of that class. It does not require a conversion to other units, since it is the class representative. All other units within the class will have a conversion to this base unit. This element is of type baseUnitType.
>
> **sequence:** The other choice is the following sequence. The sequence must be one or the other or both of the next two elements: unknown flag and/or ConversionToBaseUnit.
>
> > **element – unknown:** unknown is an empty tag (no content). If present, it indicates that the meaning of the unit of measure is not known. If present, it is also possible to give a conversion to a base unit, although it is understood that the conversion is a "best guess" and should be suspect. The main use of the unknown flag is expected to occur with legacy data for which a unit symbol is given, but the exact meaning of the unit is not known. For example, a unit symbol of "psi" in legacy data probably means pounds per square inch, but it is unknown whether it is absolute or gauge pressure.
> >
> > **element –ConversionToBaseUnit:** This element defines the parameters that allow the conversion to a base unit. It is of type "conversionType", which will be described below.

attribute – uid: The value of the mandatory uid attribute is used so that other instances may reference this unit definition. It is type ID, and must be unique within the XML file.

attribute – annotation: The annotation attribute is optional. It is given to assist applications in the plotting or printing of the values and their units of measure. The value of the annotation attribute is the string that may be used on a plot or print. It is included at this point to allow consistency within the file, but it is considered as a hint, which may be overridden by an application.

Class: unitDictionaryEntry

```
<xsd:complexType name="unitDictionaryEntry">
  <xsd:sequence>
    <xsd:element name="name" type="xsd:string"/>
    <xsd:element name="measurementType" type="xsd:string"
         minOccurs="0"/>
    <xsd:element name="catalogName" type="xsd:string" minOccurs="0"/>
    <xsd:element name="catalogSymbol" type="explicitType"
         minOccurs="0"/>
    <xsd:choice>
      <xsd:element name="BaseUnit" type="baseUnitType"/>
      <xsd:element name="ConversionToBaseUnit"
         type="conversionType"/>
    </xsd:choice>
  </xsd:sequence>
```

```
    <xsd:attribute name="uid" type="xsd:ID" use="required"/>
    <xsd:attribute name="annotation" type="xsd:string" use="optional"/>
  </xsd:complexType>
```

An element of type unitDictionaryEntry is intended to be used in a controlled Units Dictionary as described in the patterns of usage section. It is very close to the datatype, unitDefinitionType. It will not be described in detail, but will be contrasted with the unitDefinitionType as described earlier.

It should be noted that the display element can also be added to this class. See Appendix D, display element option.

The **name** is mandatory. All other subelements are the same as in unitDefinitionType.

The **choice** does not allow the unknown flag. The purpose of the unknown flag is to indicate that a particular unit of measure included in a file has an unknown conversion to a base unit. This information does not make sense in a dictionary file. It is expected that any entry in a units dictionary will be a well-defined unit of measure – either as a base unit or with a conversion to a base unit.

Complex Datatype: explicitType

```
  <xsd:complexType name="explicitType">
    <xsd:simpleContent>
      <xsd:extension base="xsd:string">
        <xsd:attribute name="explicit" use="optional">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:enumeration value="Yes"/>
              <xsd:enumeration value="No"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:attribute>
      </xsd:extension>
    </xsd:simpleContent>
  </xsd:complexType>
```

An element of type explicitType will have a value and an attribute, explicit. This type is used within this document to define the element, catalogSymbol. The meaning of catalogSymbol, and the explicit attribute, is given in description of the complex type, unitDefinitionType.

Class: baseUnitType

```
  <xsd:complexType name="baseUnitType">
   <xsd:sequence>
    <xsd:element name="description" type="xsd:string" minOccurs="0"/>
    <xsd:element name="basicAuthority" type="xsd:string"
         minOccurs="0"/>
   </xsd:sequence>
  </xsd:complexType>
```

An element of type baseUnitType is used to define the unit of measure as being a base unit. The element may be empty, but may optionally contain two subelements, description and basicAuthority. If empty, it may be considered as a flag that defines the unit as being a base unit – the representative of the particular units equivalence class.

element – description: Description is an uncontrolled string value that may be used to give an informative description of the unit of measure.

element – basicAuthority: basicAuthority is an uncontrolled string value that may be used to give the authority that originally defined the unit of measure. In general, this will be ISO 1000, but it may be another authority. For example, the American Petroleum Institute defines the base unit, API gamma ray units.

Class: conversionType

```
<xsd:complexType name="conversionType">
  <xsd:sequence>
    <xsd:choice>
      <xsd:element name="factor" type="xsd:double"/>
      <xsd:sequence>
        <xsd:element name="numerator" type="xsd:double"/>
        <xsd:element name="denominator" type="xsd:double"/>
      </xsd:sequence>
      <xsd:sequence>
        <xsd:element name="firstTerm" type="xsd:double"
           minOccurs="0"/>
        <xsd:element name="secondTerm" type="xsd:double"/>
        <xsd:element name="thirdTerm" type="xsd:double"/>
        <xsd:element name="fourthTerm" type="xsd:double"
           minOccurs="0"/>
      </xsd:sequence>
    </xsd:choice>
    <xsd:element name="description" type="xsd:string" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="baseUnit" type="xsd:anyURI" use="required"/>
</xsd:complexType>
```

An element of type conversionType is a class that encapsulates the linear rational model for units conversion. Since the model is explained elsewhere in this document, this section will explain how the subelements match up to the formula, Y = (A + BX)/(C + DX).

In addition to a choice of three options for presenting the conversion parameters, the class also includes the subelement, description, which is intended to give an informative comment on the conversion process.

choice:

> **element – factor:** A = D = 0, C = 1, B = factor. An example is the conversion from the international foot to metres, for which factor = .3048.
>
> **elements - numerator, denominator:** A = D = 0, B = numerator, C = denominator. An example is the conversion from the US Survey foot to metres, for which numerator = 12., denominator = 39.37.
>
> **elements – firstTerm, secondTerm, thirdTerm, fourthTerm:** These correspond respectively to A, B, C, and D in the formula.

element – description: An optional and uncontrolled string value. This is intended to be a free form comment on the conversion process. It is not intended to be interpreted by software.

Appendix B: Example of Units Dictionary

The Units Dictionary contained here includes an example of a Document Information Class that is not part of the recommendation.

(The namespace strategy is elementFormDefault=”undefined”. This strategy is not part of the recommendation. At this point, no namespace strategy is being recommended.)

It is expected that each local site will develop its own Document Information Class.

The Units Dictionary essentially comprises instances of the unitDictionaryEntry Class.

XML Sample:

```
<?xml version="1.0"?>
<pr:UnitOfMeasureDictionary>
       xmlns="http://www.posc.org/schemas"
       xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance"
       xmlns:pr="http://www.posc.org/applications"
       xsi:schemaLocation="http://www.posc.org/applications
                  http://www.posc.org/applications/unitsdict.xsd">

  <DocumentInformation>
    <name>Units Dictionary for EPSG Units of Measure</name>
    <version>0.1 sample</version>
    <description>Sample of the EPSG units for geodetic information
put into a dictionary. This is for illustration only.</description>
    <creator associatedWith="POSC">John Bobbitt</creator>
    <creationDate>
      <isoDate>2001-03-23</isoDate>
    </creationDate>
  </DocumentInformation>

  <UnitsDefinition>
    <UnitOfMeasure uid="m" annotation="m">
      <name>metre</name>
      <measurementType>length</measurementType>
      <catalogName>EPSG abbreviation</catalogName>
      <catalogSymbol explicit="Yes">m</catalogSymbol>
      <BaseUnit>
        <description>The metre is the length equal to 1 650 763.73
wavelengths in vacuum of the radiation corresponding to the
transition between the levels 2p10 and 5d5 of the krypton-86
atom.</description>
        <basicAuthority>ISO 1000</basicAuthority>
      </BaseUnit>
    </UnitOfMeasure>

    <UnitOfMeasure uid="rad" annotation="rad">
      <name>radian</name>
      <measurementType>angle</measurementType>
      <catalogName>EPSG abbreviation</catalogName>
      <catalogSymbol explicit="Yes">rad</catalogSymbol>
      <BaseUnit>
        <description>The radian is the plane angle between two radii
of a circle that cut of, on the circumference, an arc equal in length
to the radius.</description>
        <basicAuthority>ISO 1000</basicAuthority>
      </BaseUnit>
    </UnitOfMeasure>

    <UnitOfMeasure uid="ft" annotation="ft">
      <name>foot</name>
      <measurementType>length</measurementType>
      <catalogName>EPSG abbreviation</catalogName>
      <catalogSymbol explicit="Yes">ft</catalogSymbol>
      <ConversionToBaseUnit baseUnit="#m">
        <factor>.3048</factor>
      </ConversionToBaseUnit>
    </UnitOfMeasure>

    <UnitOfMeasure uid="ftUS" annotation="US ft">
      <name>US survey foot</name>
      <measurementType>length</measurementType>
      <catalogName>EPSG abbreviation</catalogName>
      <catalogSymbol explicit="Yes">ftUS</catalogSymbol>
      <ConversionToBaseUnit baseUnit="#m">
        <numerator>12</numerator>
        <denominator>39.37</denominator>
      </ConversionToBaseUnit>
    </UnitOfMeasure>

    <UnitOfMeasure uid="ftCla" annotation="ftCla">
      <name>Clarke's foot</name>
      <measurementType>length</measurementType>
      <catalogName>EPSG abbreviation</catalogName>
      <catalogSymbol explicit="Yes">ftCla</catalogSymbol>
      <ConversionToBaseUnit baseUnit="#m">
        <factor>0.304797265</factor>
      </ConversionToBaseUnit>
    </UnitOfMeasure>

    <UnitOfMeasure uid="fathom" annotation="fathom">
      <name>fathom</name>
      <measurementType>length</measurementType>
      <catalogName>EPSG abbreviation</catalogName>
      <catalogSymbol explicit="Yes">f</catalogSymbol>
      <ConversionToBaseUnit baseUnit="#m">
        <factor>1.8288</factor>
      </ConversionToBaseUnit>
    </UnitOfMeasure>

    <UnitOfMeasure uid="nmi" annotation="NM">
      <name>Nautical Mile</name>
      <measurementType>length</measurementType>
      <catalogName>EPSG abbreviation</catalogName>
      <catalogSymbol explicit="Yes">NM</catalogSymbol>
      <ConversionToBaseUnit baseUnit="#m">
        <factor>1852</factor>
      </ConversionToBaseUnit>
    </UnitOfMeasure>

    <UnitOfMeasure uid="glm" annotation="M Ger">
      <name>German Legal Metre</name>
      <measurementType>length</measurementType>
      <catalogName>EPSG abbreviation</catalogName>
      <catalogSymbol explicit="Yes">GLM</catalogSymbol>
      <ConversionToBaseUnit baseUnit="#m">
        <factor>1.000013597</factor>
      </ConversionToBaseUnit>
    </UnitOfMeasure>

    <UnitOfMeasure uid="chUS" annotation="ch">
      <name>US Survey Chain</name>
      <measurementType>length</measurementType>
      <catalogName>EPSG abbreviation</catalogName>
      <catalogSymbol explicit="Yes">chUS</catalogSymbol>
      <ConversionToBaseUnit baseUnit="#m">
        <numerator>792</numerator>
        <denominator>39.37</denominator>
      </ConversionToBaseUnit>
    </UnitOfMeasure>

    <UnitOfMeasure uid="lkUS" annotation="lk">
      <name>US Survey Link</name>
      <measurementType>length</measurementType>
      <catalogName>EPSG abbreviation</catalogName>
      <catalogSymbol explicit="Yes">lkUS</catalogSymbol>
      <ConversionToBaseUnit baseUnit="#m">
        <numerator>7.92</numerator>
        <denominator>39.37</denominator>
      </ConversionToBaseUnit>
    </UnitOfMeasure>

     <UnitOfMeasure uid="miUS" annotation="mi">
      <name>US Survey Mile</name>
      <measurementType>length</measurementType>
      <catalogName>EPSG abbreviation</catalogName>
      <catalogSymbol explicit="Yes">miUS</catalogSymbol>
      <ConversionToBaseUnit baseUnit="#m">
        <numerator>63360</numerator>
        <denominator>39.37</denominator>
      </ConversionToBaseUnit>
    </UnitOfMeasure>

    <UnitOfMeasure uid="km" annotation="km">
      <name>kilometre</name>
      <measurementType>length</measurementType>
      <catalogName>EPSG abbreviation</catalogName>
      <catalogSymbol explicit="Yes">km</catalogSymbol>
      <ConversionToBaseUnit baseUnit="#m">
        <factor>1000</factor>
      </ConversionToBaseUnit>
    </UnitOfMeasure>

  </UnitsDefinition>

</pr:UnitOfMeasureDictionary>
```

Appendix C: Sample XML File

Following is a sample XML file that illustrates the Pattern of Usage.

```
<?xml version="1.0"?>
<pr:RootElement
       xmlns="http://www.posc.org/schemas"
       xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance"
       xmlns:pr="http://www.posc.org/applications"
       xsi:schemaLocation="http://www.posc.org/applications
                  http://www.posc.org/applications/myown.xsd">
```

```
    <distanceFromWell uom="#m">899</distanceFromWell>
    <depthOfWell uom="#ft1">12994</depthOfWell>
    <leaseLength uom="#ft2">987.33</leaseLength>
    <leaseWidth uom="#ft2">287.44</leaseWidth>
    <areaOfCoverage uom="#acre">160</areaOfCoverage>
    <distanceFromBoundary uom="#vara">79.3</distanceFromBoundary>
    <CoordinateSystemAxis>
      <axisUomRef uom="#m"/>
      <axisName>X</axisName>
    </CoordinateSystemAxis>

  <UnitOfMeasureBlock>
    <uomReference
       uid="m"
       To="http://www.posc.org/applications/unitsDict.xml#m"/>
    <uomReference
       uid="ft1"
       To="http://www.posc.org/applications/unitsDict.xml#ft"/>
    <uomReference
       uid="ft2"
       To="http://www.posc.org/applications/unitsDict.xml#ftUS"/>
    <uomReference
       uid="acre"
       To="http://www.goober.com/unitsDictionary.xml#acre"/>
    <UnitOfMeasure
       uid="vara" annotation="vara">
      <name>Texas vara (modern)</name>
      <ConversionToBaseUnit baseUnit="#m">
        <factor>.8466</factor>
      </ConversionToBaseUnit>
    </UnitOfMeasure>
  </UnitOfMeasureBlock>

  </pr:RootElement>
```

Appendix D: Options

Following is an informative section. This lists options and alternatives that should be considered. In many cases, these options were rejected from the recommendations in the previous sections. This section is useful because it comments on ways that the units of measure patterns can be used to satisfy specific user needs. It also comments on variations that would probably not be interoperable if they were to be used.

Option: Currencies.

The suggestion was made to allow the quantityType to include currencies. This was rejected for two reasons:

1. Currencies are not convertible to a “base” currency. The conversion factor changes from day to day. Thus, the definition portion is not valid.
2. There is a well-defined list of currency symbols, developed and maintained by ISO. Given this well-defined list, it is possible to unambiguously reference a currency using its symbol.

However, it is possible for the currency data type to “look” like the quantity data type.

XML Example:

```
<finalCost currency=”USD”>98000</finalCost>
```

XML Schema:

```
<!--Sample Usage-->
<xsd:element name=”finalCost” type=”my:moneyType”/>
<!--abbreviated Schema for moneyType-->

<xsd:complexType name="moneyType">
  <xsd:simpleContent>
    <xsd:extension base="xsd:decimal">
      <xsd:attribute name="currency" type="my:currencies"/>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>

<xsd:simpleType name="currencies">
  <xsd:restriction base="xsd:string">
   <xsd:enumeration value="ADF"/>
   <xsd:enumeration value="ADP"/>
   <xsd:enumeration value="AED"/>
   <xsd:enumeration value="AFA"/>
       ...
   <xsd:enumeration value="ZWD"/>
  </xsd:restriction>
</xsd:simpleType>
```

Option: No skip reference

The comment has been made that the skip reference adds extra lines that serve no purpose. It would be more efficient to give the URI when the unit of measure is used, rather than giving a local reference to a skip reference tag. Using this pattern, the XML in Appendix C would look as follows:

Alternative XML from Appendix C.

```
<?xml version="1.0"?>
<pr:RootElement
       xmlns="http://www.posc.org/schemas"
       xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance"
       xmlns:pr="http://www.posc.org/applications"
       xsi:schemaLocation="http://www.posc.org/applications
                  http://www.posc.org/applications/myown.xsd">

  <distanceFromWell
     uom="http://www.posc.org/applications/unitsDict.xml#m">
              899</distanceFromWell>
  <depthOfWell
     uom="http://www.posc.org/applications/unitsDict.xml#ft">
              12994</depthOfWell>
  <leaseLength
     uom="http://www.posc.org/applications/unitsDict.xml#ftUS">
              987.33</leaseLength>
  <leaseWidth
     uom="http://www.posc.org/applications/unitsDict.xml#ftUS">
              287.44</leaseWidth>
  <areaOfCoverage
     uom="http://www.goober.com/unitsDictionary.xml#acre">
```

```
                160</areaOfCoverage>
    <distanceFromBoundary uom="#vara">
                79.3</distanceFromBoundary>

  <UnitOfMeasureBlock>
  <UnitOfMeasure uid="vara" annotation="vara">
      <name>Texas vara (modern)</name>
      <ConversionToBaseUnit
          baseUnit="http://www.posc.org/applications/unitsDict.xml#m">
        <factor>.8466</factor>
      </ConversionToBaseUnit>
    </UnitOfMeasure>
  </UnitOfMeasureBlock>

  </pr:RootElement>
```

This pattern is acceptable, but not recommended. The complications of internally vs. externally defined units of measure is pushed into the usage of the unit, rather than into the UnitOfMeasureBlock as was in the original pattern. Also, the UnitOfMeasureBlock is optional, and is needed only if there is a unit that is not in some well known dictionary. Finally, there is an advantage to collecting together all of the units of measure in one place (the UnitsOfMeasureBlock) so that the reader – whether human or software – can find all of the units more easily.

Option: Restricted Quantity Types

The suggestion has been made that there be a set of quantity types that have a restricted set of units of measure. For example, an angleQuantityType that would be restricted to radians, degrees, minutes, seconds, grads, and mils. It would look like the following:

Sample Schema for restricted angle type:

```
  <xsd:complexType name="angleQuantityType">
    <xsd:simpleContent>
      <xsd:extension base="xsd:decimal">
        <xsd:attribute name="uom" type="my:angleList"/>
      </xsd:extension>
    </xsd:simpleContent>
  </xsd:complexType>

  <xsd:simpleType name="angleList">
    <xsd:restriction base="xsd:string">
     <xsd:enumeration value="deg"/>
     <xsd:enumeration value="min"/>
     <xsd:enumeration value="sec"/>
     <xsd:enumeration value="rad"/>
     <xsd:enumeration value="grad"/>
     <xsd:enumeration value="mil"/>
    </xsd:restriction>
  </xsd:simpleType>
```

Note that this is the same pattern as the currency type mentioned earlier.

Here are the objections to this that have been raised.

1. The reference is by string, rather than URI. Thus, an application reading the file must know the meaning of the unit from an outside source.
2. A reader application must be able to handle string references for some units, and URI references for others.

3. The only way to extend the list of acceptable units is to change the schema.
4. In many cases, a file is sent with an understandable uom symbol, but one that is not in the list. For example, a unit may be declared to be DEG, or DMS. Some users have given the requirement that an unknown unit should be accepted by the file, and a user or smart application can correct it.

Option: Consider different formats to be different units

With angles, in particular, there are formats such as the sexagesimal DDD:MM:SS.sss. This recommendation takes the view that these are not different units of measure. Rather, they are different formats.

Here is the same angle in several different formats:

122:38:12.44 W

-122:38:12.44

-122.6367889 degrees

The first two are the sexigesimal form – one with a sign, and the other with a hemisphere designation – while the third is a quantity form. This paper takes the view that only the third is to be expressed as a quantity data type. The other two may use something like an angle data type. Here are samples of the three in XML where the schema introduces a tag to distinguish between them.

XML Sample of Angles

```
<longitude>
  <dms>
    <degree direction=”-“>122</degree>
    <minute>38</minute>
    <second>12.44</second>
  </dms>
</longitude>

<longitude>
  <dms>
    <degree direction=”W“>122</degree>
    <minute>38</minute>
    <second>12.44</second>
  </dms>
</longitude>

<longitude>
  <quantityFormat uom=”#dega”>-122.6367889</quantityFormat>
</longitude>
```

The only format that needs units of measure specified is the quantity format, because the sexagessimal format has units specified within the format itself.

Option: Key on symbols rather than URI Reference (1)

Note: There are two cases for this option. The second is given immediately following.

A few have expressed a desire to have the unit of measure reference a well-known symbol rather than a URI that defines the symbol. The argument is that there are standard dictionaries of symbols that can be used by a wide group of applications.

However, as pointed out in many cases, there is no comprehensive dictionary. Virtually every standard dictionary has missing units. Also, there are standard dictionaries that use the same symbol for different units – sometimes through a combining of prefixes, other times through a reuse of a symbol. Examples are:

- min can be a symbol for minute (time), minute (angle), or milli-inch.
- nm can be a symbol for nanometre or nautical mile.
- A, a, and a are defined in ISO 1000 as Ampere, year (annum), and are (areal measure).

The above examples become more compelling when the symbol is considered case insensitive.

Given the problems inherent in finding a comprehensive set of symbols, it has been recommended by most contributors to this recommendation that the uri reference, with unit definition or unit library by used.

However, the following method is recommended for those who wish to reference by symbol.

1. Use the same quantityType as recommended in this paper. (with uom of type anyURI).
2. Add a “skip reference” type element that used a symbol (string) rather than an anyURI:

```
   <epsgSymbolRef uid=”#ft” uomSymbol=”ftUS”/>
```
where this tag would appear in the units definition block. To accomplish this, the unitBlockEntry type would be changed to the following.

```
   <complexType name="uomBlockEntryExtended">
     <choice maxOccurs="unbounded">
       <element name="uomReference" type="my:refToType"/>
       <element name="UnitOfMeasure" type="my:unitDefinitionType"/>
       <element name="epsgSymbolRef" type="my:SymbolRef"/>
     </choice>
   </complexType>
   <xsd:complexType name=”symbolRef”>
     <xsd:attribute name=”uid” type=”xsd:ID” use=”required”/>
     <xsd:attribute name=”To” type=”xsd:string” use=”required”/>
   </xsd:complexType>
```
A further refinement could be made by enumerating all the symbols that are acceptable. This would insure that only known symbol abbreviations would be used.

Option: Key on symbols rather than URI Reference (2)

A second need that has been expressed is that there be no units block. This is particularly true in the case of a real-time application that does not have a well defined section in which the units block can be written. The request is that the usage of the unit of measure be referenced by symbol rather than by anyURI. Here is an example of the XML:

```
   <gammaRay uom=”gAPI”>27.2</gammaRay>
   <resistivity uom=”ohm.m”>43.8</resistivity>
```

This would be accomplished with the following definition for a quantity datatype:

```
   <xsd:complexType name=”symbolQuantityType”>
     <xsd:simpleContent>
       <xsd:extension base=”xsd:double”>
         <xsd:attribute name=”uom” type=”xsd:string” use=”required”/>
       </xsd:extension>
     </xsd:simpleContent>
   </xsd:complexType>
```

The only difference between the quantityType and the (above) symbolQuantityType is that the latter uses a string for the uom value, while the former uses an anyURI.

This datatype only makes sense if the symbol is understood by the reading program. For this reason, there must be certain requirements for its successful use:

1. The symbol must be in a well-known dictionary.

2. The dictionary must be defined for the use, either by general agreement or by a tag within the XML file.
3. Any reading application must know and understand the unit of measure symbol being used. This imposes a strong requirement on the reading application. In practice, it means that only a limited number of reading applications will ever reference the file. The file is not interoperable outside of the limited context.

If the usage case for the XML file meets the above requirements, there is no reason why this cannot be a succesful strategy, that is "close to" the recommendations of this paper. However, it must be understood that it is non-conformant for general usage.

Options: Alternative Units Models.

The recommendations above do not specify either of the two basic units models, nor do they attempt to define any operations inherent in these models.

Units of measure can be formed by (algebraic) combinations of units to form other units. In ISO 1000, these are referred to as derived units of measure, some of which are given special symbols, but many of which use the algebraic symbol combinations. Other models consider these derived units to be merely symbols for the unit of measure itself.

Consider, for example the units metre [m] for length[3], second [s] for time, and the combination metre per second [m/s] for velocity. The "algebra of units" model would understand that the m/s is a combination of the length unit divided by the time unit. The "symbol only" model would not have that understanding, and would consider m/s as yet another symbol, this for metres per second, which is a unit of velocity.

Both methods are valid. Both have advantages and disadvantages. Both are supported in the above recommendations.

The point where the two differ is in the unitDictionaryEntry or unitDefinitionType Class catalogSymbol element. In that element, there is an attribute, explict = "Yes" or "No", which is used to state whether the cataog being referenced explicitly carries the symbol (m/s in this case), or if it is implied (m is there, s is there, and the algebra implies that m/s can be understood).

This leaves the choice of the units model out of the realm of this recommendation. It is not the purpose of this recommendation to specify a particular units model, but to give the information, and allow any unit model to be used.

Option: Alternative Units of Measure Conversion Formulas

The issue is how to group various units of measure into meaningful equivalence classes. In this case, an equivalence class is a set of units that are convertible amongst themselves, based on a particular formula (the linear rational transformation: $Y = (A + BX)/(C + DX)$). Within an equivalence class, one member is chosen as its representative (the base unit), and the others are considered to be conversions from that base unit.

Suggestions have been made that the conversion formula be extended so that equivalence classes (with units such as Bels) can be grouped with other equivalence classes (in this case with units of power).

The linear rational transformation was chosen because it allows a general enough grouping of units, yet is easy to implement.

The methods may be extended by defining another "choice" in the Conversion class, and defining the meanings of the paramaters that are encapsulated in that choice.

Here are the implications of the linear rational model.

[3] The [xxx] notation indicates the symbol for the unit of measure. [xxx] would indicate that the symbol is 'xxx'.

Many units transformation formulas use a factor, only. This corresponds to the B term (with A = D = 0, and C = 1.0). However, implementors have found that the C term is useful to maintain enough significant digits. For example, conversion from kilometres per hour to metres per second is accomplished with B = 1000, and C = 3600. The factor (1000/3600), would be a decimal, rounded off, with a possible loss of precision.

The A term allows (for example) temperature units to be in the same class, because temperature units have an offset term in their transformation. It also allows gauge pressure and absolute pressure to be in the same class.

The D term allows only one case that I know of. That is the case of api Gravity being classified with the density units. 30 weight oil, for example, would convert to .876 grams per cubic centimetre.

As pointed out, this formula does not allow the power ratio class to be combined with the power class. Thus, it is necessary to define a power ratio class with a base unit of Bels, and an alternative unit of deciBels (and other alternative units, if needed).

It is felt that the linear, rational model is a good choice in balancing the intuitive concepts of which units should be equivalenced, with practicality, and implementability.

Option: Display element

It is expected that an XML file will be viewed through an XSL stylesheet (or converted using XSLT). It is useful to insure that the quantity value along with its unit of measure can be displayed appropriately using this specification.

The annotation attribute has been attached to the units dictionary to give a display option. This is done because the annotation allows more characters than does an attribute of type xsd:ID. However, it does not add enough.

Consider the example of feet per second squared (acceleration unit). The simplest display, using the 7 bit character set, would be something like ‘ft/s2’ or ‘ft/sec2’. It should be noted that the “/” symbol was not allowed in values of type ID, although this varied considerably depending on the implementation. Thus, the ID value might be ‘fps2’, which is not meant to be used for annotation, but merely used for referencing.

Many consider the two choices, ‘ft/s2’ and ‘ft/sec2’, to be unsuitable. Rather they would like to see ‘ft/sec^2 in the display of the unit of measure. This corresponds to the HTML form of

```
ft/sec<sup>2</sup>
```

We can capture this option by adding an element to the unitDictionaryEntry and the unitDefinitionClass:

```
<xsd:element name=”display” minOccurs=”0” type=”displayType”/>

<xsd:complexType name=”displayType” mixed=”true”?
  <xsd:choice minOccurs=”0” maxOccurs=”unbounded”>
    <xsd:element name=”sup” type=”xsd:string”/>
    <xsd:element name=”sub” type=”xsd:string”/>
  </xsd:choice>
</xsd:complexType>
```

This allows the units definitions to carry a more meaningful display choice. The entry for feet per second squared would look like the following:

```
<UnitOfMeasure uid="ftps2" annotation="ft/s2">
  <name>feet per second per second</name>
  <measurementType>acceleration</measurementType>
  <display>ft/sec<sup>2</sup></display>
  <ConversionToBaseUnit baseUnit="#mps2">
    <factor>.3048</factor>
  </ConversionToBaseUnit>
</UnitOfMeasure>
```

An XSL snippet to handle this might look like the following:

```
<xsl:template match="p:UnitOfMeasure">
  <xsl:choose>
    <xsl:when test="p:display">
      <xsl:apply-templates select="p:display"/></xsl:when>
    <xsl:when test="@annotation">
      <xsl:value-of select="@annotation"/></xsl:when>
    <xsl:when test="p:catalogSymbol">
      <xsl:value-of select="p:catalogSymbol"/></xsl:when>
    <xsl:otherwise>
      <xsl:value-of select="@uid"/></xsl:otherwise>
  </xsl:choose>
</xsl:template>

<xsl:template match="p:sup">
  <sup><xsl:apply-templates/></sup>
</xsl:template>

<xsl:template match="p:sub">
  <sub><xsl:apply-templates/></sub>
</xsl:template>
```