International
Virtual
Observatory
Alliance

Astronomical Measurements Model Version 1.0

IVOA Working Draft 2019-05-01

Working group
Data Model Working Group

This version
http://www.ivoa.net/documents/Measurements/20190501

Latest version
http://www.ivoa.net/documents/Measurements

Previous versions
This is the first public release

Author(s)
Arnold Rots, Mark Cresitello-Dittmar

Editor(s)
Mark Cresitello-Dittmar, Arnold Rots

Abstract

In creating version 2 of "*Space-Time Coordinate Metadata for the Virtual Observatory*" (STC) (Rots, 2007) Data Model, it was decided to split the content into various component models which focus on particular aspects of the previous model scope.

This model covers the description of measured or determined astronomical data, and includes the following concepts:

- The association of the determined 'value' with corresponding errors. In this model, the 'value' is given by the various Coordinate types of the Coordinates data model (Rots and Cresitello-Dittmar et al., 2019).
- A description of the Error model.

Status of this document

This is an IVOA Working Draft for review by IVOA members and other interested parties. It is a draft document and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use IVOA Working Drafts as reference materials or to cite them as other than "work in progress".

A list of current IVOA Recommendations and other technical documents can be found at http://www.ivoa.net/documents/.

Contents

Acknowledgments

TBD: Get appropriate acknowledgment text

Conformance-related definitions

The words "MUST", "SHALL", "SHOULD", "MAY", "RECOMMENDED", and "OPTIONAL" (in upper or lower case) used in this document are to be interpreted as described in IETF standard RFC2119 (Bradner, 1997).

The *Virtual Observatory (VO)* is a general term for a collection of federated resources that can be used to conduct astronomical research, education, and outreach. The International Virtual Observatory Alliance (IVOA) is a global collaboration of separately funded projects to develop standards and infrastructure that enable VO applications.

1 Introduction

1.1 Motivation

The collection and analysis of observed or derived data is the foundation of Astronomy. These data form the core of our most fundamental products:

- Observational data - raw data in detector coordinates and instramental units.
- Derived data products - images and tables of derived data in physical units and coordinate spaces.
- Simulated data - data derived from simulation of physical entities and instrumentation.
- Catalogs - collections properties associated with various astronomical sources derived from multiple observations of that source.

Performing astronomical data analysis in the Virtual Observatory means to discover, evaluate, extract, combine and manipulate data which may have been obtained at different times, with different instruments, and different locations. To confidently execute this workflow requires a complete and accurate description of the nature of the data and its associated errors.

The "*Astronomical Coordinates and Coordinate Systems*" (Coords) (Rots and Cresitello-Dittmar et al., 2019) Data Model provides the description of the nature of the data values, the Coordinate space, reference frame, domain, etc. Here, we utilize that information to build a model for observed or derived data which adds elements quantifying the accuracy of the data.

1.2 Context and Scope

This document results from updating the "*Space-Time Coordinate Metadata for the Virtual Observatory*" (STC) (Rots, 2007) model for use in VO-DML compliant models. That model provides metadata for describing Space-Time related, and other Coordinates including associated errors. These metadata are to be used for specifying coordinate-related information for datasets, catalogs, and queries.

The update and revision of the STC model has sub-divided the content into component models, each covering a portion of the scope of the parent model. This allows for better description of the relations between the various components, allows for independent development of the component models, and creates smaller, more digestible content for users.

This document describes the Measurement model which provides descriptions for:

- The observed or derived data 'value'.
- The error(s) associated with that 'value'.

2 Use Cases and Requirements

2.1 Use Cases

2.1.1 Cube model support

The primary use case for this work is in support of the CubesDM.
The CubesDM is a N-Dimensional model for pixelated images and sparse cube data.

The following is a brief outline of the most relevant features pertaining to the development of the Measurement, Coordinates, and Transform component models. Many of these features

are handled by the Coordinates or Transform models, but are relevant to the full description of Measurement model elements.

- General
  - knowledge of the physical domain spaces provided at a high level
  - definition of the domain space includes the following criteria
    - dimensionality (typically 1,2 or 3 for physical domain), pixel domain may be of any dimension
    - axis configuration (for spatial domain which has >1D). The most common configurations for astronomical data are Cartesian and Spherical, but others may be used as well.
    - domain range along each axis, typically +/- Inf, but may be limited due to physical constraints (e.g. physical size of a detector, sensitivity limitiations, etc)
    - association with additional metadata further describing the nature of the domain space (Frame). This is especially true for the Spatial and Temporal domains, but may apply to others as well. Examples include:
      - reference position (location of origin)
      - reference frame (orientation of the domain space)
      - planetary ephemeris
      - equinox
- Pixelated Image Cube
  - complete specification of pixel coordinate domain; number of axes, number of pixels per axis
  - image axes
    - in pixel domain, are a binned coordinate space with integerized values (pixel indexes)
    - mapped to various 'physical' coordinate spaces via transform operations
    - mappings may define a progressive migration in coordinate space (e.g. pixel - ccd - detector - sky - wcs)
    - pixel axis mappings are typically to a continuous domain, but may also be to a discrete domain such as Polarization state.
  - image cubes may have any number of dimensions, but are typically separable into co-dependent axes of 1, 2, or 3 dimensions.
    - spatial domain typically 2-3 dimensions
    - other domains (time, spectral, polarization), are typically 1 dimensional
  - image data value is typically given in a physical domain, but may itself be mapped to other domains
- Sparse Cube
  - data axes cover a wide array of physical domains including, but not limited to Spatial, Temporal, Spectral, Polarization,
  - individual domains may be represented multiple times in different frames (ccd, detector, sky; pha, energy)

  - data values may have associated errors
    - typical error forms include: symmetric(+/- a), asymmetric(+a:-b), interval (a:b), matrix, elliptical
    - can become quite complex: probability distributions, error maps, etc.
    - quality indicators:
      - global status, typically numeric
      - bit array, where each bit is associated with a particular quality state
    - associated errors may be separable or correlated among multiple data axes
  - data axes may be virtual, defined as a mapping from other data axes (same description as above)

- Physical Data (Observables)
  - focus on the following domains which are frequently included in astronomical data cubes: Spatial, Spectral, Temporal, Polarization
  - Spatial
    - Cartesian space: chip, detector, sky
    - Spherical space: Equatorial, Ecliptic, Galactic, LongLat
  - Time
    - 1-Dimensional: JD, MJD, ISOTime, TimeOffset
  - Polarization
    - Discrete space: Polarization states (Stokes, Linear, Circular, Vector)
  - Spectral
    - 1-Dimensional: energy, frequency, wavelength

2.2 Requirements

Examination and implementation of the above cases leads to a set of requirements distributed through the various STC component models. Here we itemize those relevant to this model specifically.

2.2.1 General

Requirements pertaining to the overall criteria that the model must satisfy.

[vodml.001]: The model shall be vo-dml compliant

[vodml.002]: shall re-use, or refer to, dependent models for objects and concepts already defined in other models

[vodml.003]: shall produce a validated vo-dml XML description

[vodml.004]: shall produce documentation in vo-dml HTML format

[vodml.005]: shall produce documentation in standard PDF format

2.2.2 Application/Usage

Requirements pertaining to the user experience. Note, as a data model, users will not typically interact directly with the model,

[user.001]: Users should be able to identify and use basic content with minimal specialized information. In other words, a generic utility should be able to find and use core elements without knowing a lot about the various extensions and uses of those elements.

[user.002]: When applicable, the model should support usability by simplifying common scenarios. i.e. common things simple, complex things possible

[user.003]: The model shall be easily extended to accommodate cases and applications not yet considered.

2.2.3 Content

Requirements pertaining to the elements to be defined by the model.

- Domains

  [dom.001]: Shall accommodate the description of data in any observable domain

  [dom.002]: Shall provide enhanced/specialized description for data pertaining to

  [dom.002.1]: Pixel domain: binned, integerized, n-dimensional domain

  [dom.002.2]: Spatial domain: continuous domain, typically in 2-3 dimensional cartesian or spherical spaces

  [dom.002.3]: Time domain: continuous 1D domain, typically provided in JD, MJD, ISO, or as an Offset from a zero point

  [dom.002.4]: Polarization domain: discrete 1D domain of polarization states.

- Measurements

  [meas.001]: Shall relate a coordinate value with associated errors

  [meas.002]: Shall support multiple error associations per value to describe errors from different sources

  [meas.003]: Any specific error source may appear only once

  [meas.004]: Errors may be correlated between component values (ie: may apply to coordinate set as a whole)

  [meas.005]: Values associated with different domains may have correlated errors (ie: components of coordinate tuple may refer to different domains, and have non-separable errors)

  [meas.006]: Shall support the most common error forms, including, but not limited to: Symmetrical, Asymmetrical, Interval, Elliptical, Matrix

  [meas.007]: Shall provide specialized objects related to measurements in the priority domains (Spatial, Spectral, Temporal, Polarization); leveraging [user.0002] where possible

  [meas.008]: Shall allow for the representation data outside the priority domains

2.3 Role within the VO Architecture

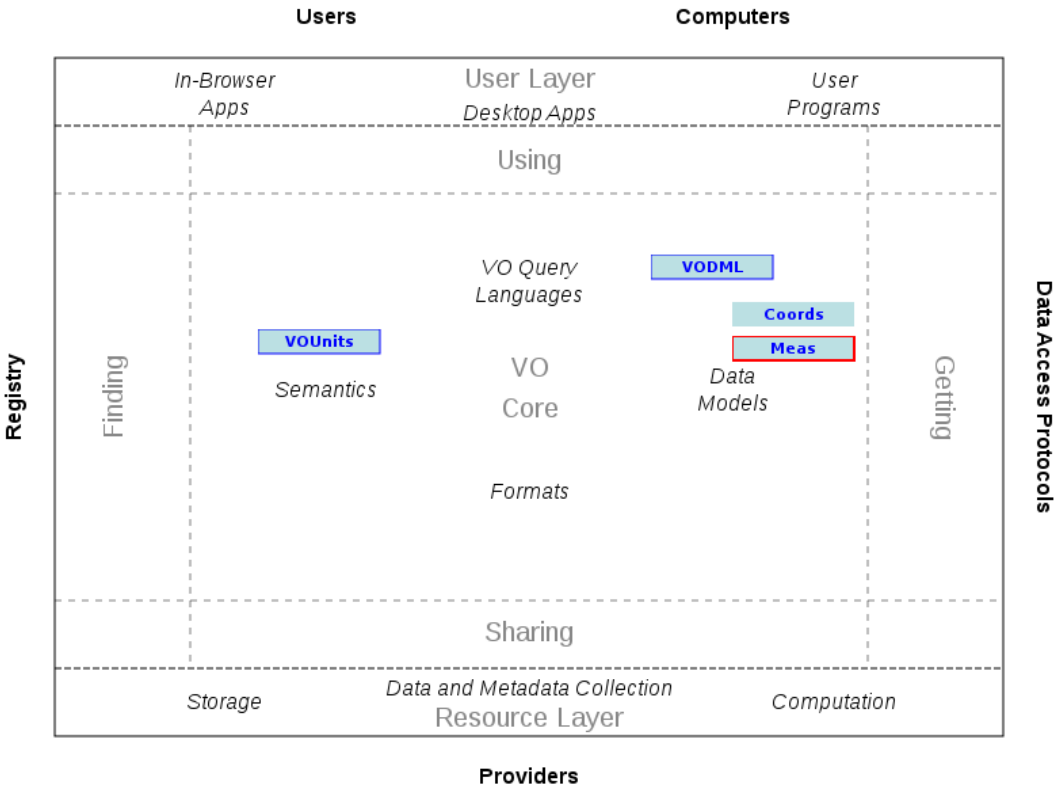

Figure 1: Architecture diagram for this document

Fig. 1 shows the role this document plays within the IVOA architecture (Arviset and Gaudet et al., 2010).

3 Model: meas

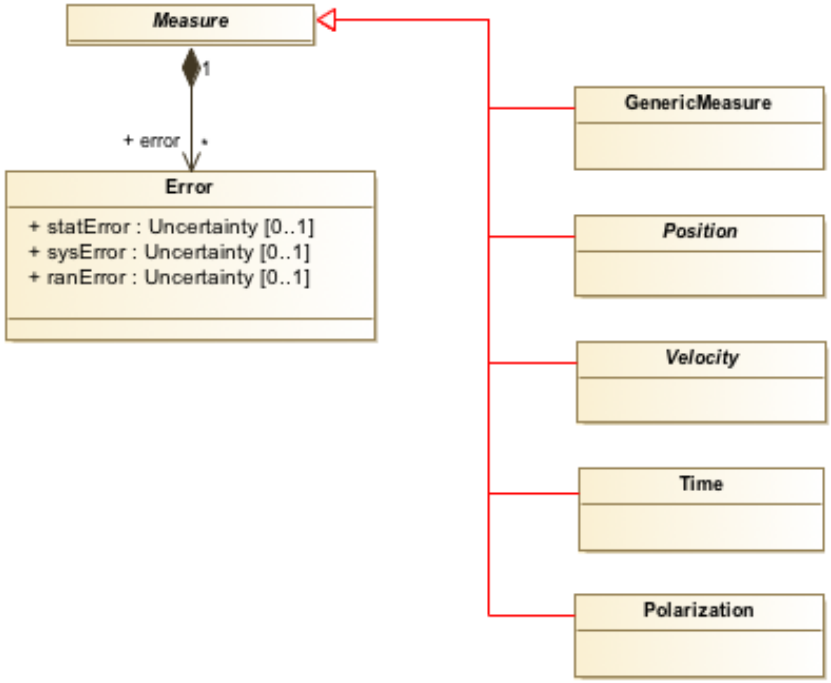

Figure 2: Overview of Measurement model elements

This model defines objects and datatypes which represent 'measured' or 'determined' data. It associates a Coordinate (ie: the determined value) with corresponding Error(s). As such, this model is at least the foundation for representing the vast majority of the Astronomical data found in catalogs and data products.

We define here, several specialized classes which cover the most basic and common types, such as Position and Time. We also provide a generic model which can accommodate virtually any data, but may require a bit more effort to fully describe the coordinate metadata. Additional specializations, e.g. Flux, Magnitude, Luminosity, Pressure, Temperature, etc. may be added to this model, or in other models focusing on particular domains or use cases.

We include a fairly simple Error model, describing errors as a 'shape' of uncertainty, and define a small set of commonly occuring forms (e.g. Symmetrical, Bounds, Ellipsoid).

4 Measure

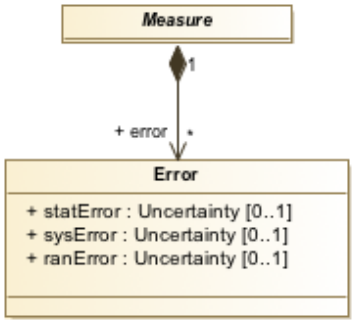

Figure 3: Top level Measure

4.1 Measure (Abstract)

Abstract base of Measure classes, associates a 'determined value' (Coordinate) with corresponding errors. We associate the Error(s) with the full Measure, rather than the individual values, in order to support both correlated and uncorrelated errors. In many cases with multi-dimensional data, the associated errors are correlated and must be considered with the value set as a whole. One consequence of this approach is that there is a looser association between the Error dimensions and the corresponding value dimension. We require that the Error content MUST be compatible with the value dimension and nature (e.g. dimension, domain, units, etc).

4.1.1 Measure.error

vodml-id: Measure.error
type: meas:Error
multiplicity: 0..*
Measurement error.

4.2 Error

The Error class uses the Uncertainty types to describe measurement errors from various sources.

4.2.1 Error.statError

vodml-id: Error.statError
type: meas:Uncertainty
multiplicity: 0..1
Statistical error. The Uncertainty type MUST be dimensionally compatible with the associated Measure value.

4.2.2 Error.sysError

vodml-id: Error.sysError
type: meas:Uncertainty

multiplicity: 0..1
Systematic error. The Uncertainty type MUST be dimensionally compatible with the associated Measure value.

4.2.3 Error.ranError

vodml-id: Error.ranError
type: meas:Uncertainty
multiplicity: 0..1
Random error. The Uncertainty type MUST be dimensionally compatible with the associated Measure value.

5 Generic Measure

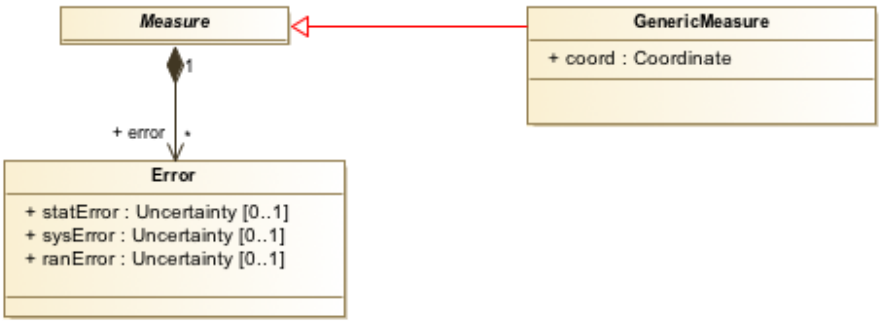

Figure 4: Generic Measure elements

5.1 GenericMeasure

The most generic Measure class. This class may be used to represent any data not covered by the specialized cases. The coordinate value type allows for the construction of multi-dimensional values to accommodate virtually any case.

5.1.1 GenericMeasure.coord

vodml-id: GenericMeasure.coord
type: coords:Coordinate
multiplicity: 1
The measured coordinate value. May be of any dimensionality (current support 1,2 and 3-dimension). For many cases, the Standard1DCoord should suffice, providing a clean and simple representation of the measured value. For other cases, users will need to provide a complete coordinate description.

6 Time

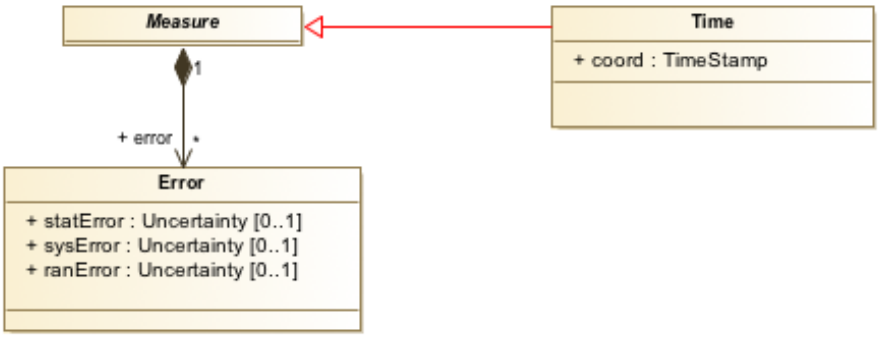

Figure 5: Time Measure elements

6.1 Time

Provides a complete description of a measured Temporal instant.

6.1.1 Time.coord

vodml-id: Time.coord
type: coords:TimeStamp
multiplicity: 1
The measured time value. May be provided in any of the TimeStamp subtypes.

7 Position

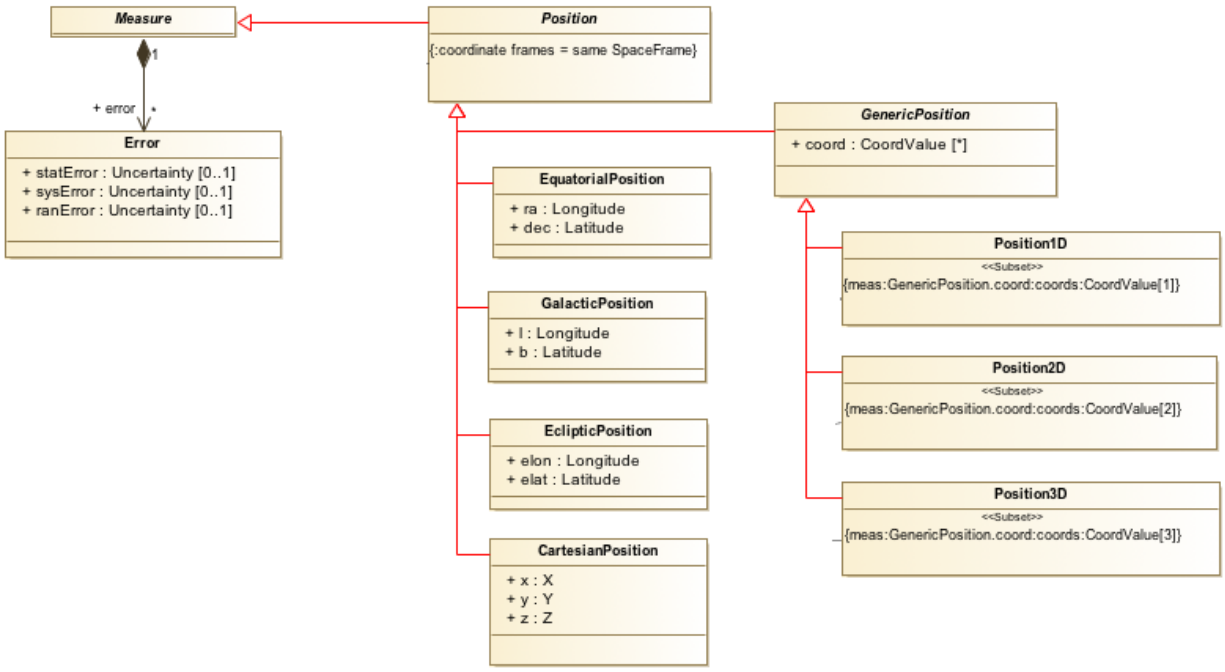

Figure 6: Position Measure elements

7.1 Position (Abstract)

Provides a complete description of a measured positional instant.
constraint
detail: Position.coordinate frames = same SpaceFrame

7.2 GenericPosition (Abstract)

Generic positional instant, appropriate for describing positional data not covered by the specialized classes. Any concrete extension of this class MUST define a specific dimensionality for the coord attribute. We provide 1,2 and 3-dimensional classes to cover the most common cases.

7.2.1 GenericPosition.coord

vodml-id: GenericPosition.coord
type: coords:CoordValue
multiplicity: 0..*
The measured position value. May be given by any CoordValue type associated with a SpaceFrame.

7.3 Position1D

Generic 1-Dimensional position.
constraint
detail: Position1D.coords:CoordValue[1]

7.4 Position2D

Generic 2-Dimensional position.
constraint
detail: Position2D.coords:CoordValue[2]

7.5 Position3D

Generic 3-Dimensional position.
constraint
detail: Position3D.coords:CoordValue[3]

7.6 EquatorialPosition

2-Dimensional position in an equatorial coordinate space. All contained coordinates MUST be associated with the same SpaceFrame instance.

7.6.1 EquatorialPosition.ra

vodml-id: EquatorialPosition.ra
type: coords:Longitude
multiplicity: 1
Right Ascension. The corresponding SpaceFrame MUST specify an equatorial reference frame, with appropriate reference position and equinox.

7.6.2 EquatorialPosition.dec

vodml-id: EquatorialPosition.dec
type: coords:Latitude
multiplicity: 1
Declination. The corresponding SpaceFrame MUST specify an equatorial reference frame, with appropriate reference position and equinox.

7.7 GalacticPosition

2-Dimensional position in an galactic coordinate space. All contained coordinates MUST be associated with the same SpaceFrame instance.

7.7.1 GalacticPosition.l

vodml-id: GalacticPosition.l
type: coords:Longitude
multiplicity: 1
Galactic Longitude. The corresponding SpaceFrame MUST specify an galactic reference frame, with appropriate reference position.

7.7.2 GalacticPosition.b

vodml-id: GalacticPosition.b
type: coords:Latitude
multiplicity: 1
Galactic Latitude. The corresponding SpaceFrame MUST specify an galactic reference frame, with appropriate reference position.

7.8 EclipticPosition

2-Dimensional position in an ecliptic coordinate space. All contained coordinates MUST be associated with the same SpaceFrame instance.

7.8.1 EclipticPosition.elon

vodml-id: EclipticPosition.elon
type: coords:Longitude
multiplicity: 1
Ecliptic Longitude. The corresponding SpaceFrame MUST specify an ecliptic reference frame, with appropriate reference position.

7.8.2 EclipticPosition.elat

vodml-id: EclipticPosition.elat
type: coords:Latitude
multiplicity: 1
Ecliptic Latitude. The corresponding SpaceFrame MUST specify an ecliptic reference frame, with appropriate reference position.

7.9 CartesianPosition

3-Dimensional position in an cartesian coordinate space.

7.9.1 CartesianPosition.x

vodml-id: CartesianPosition.x
type: coords:X
multiplicity: 1
X-Axis value.

7.9.2 CartesianPosition.y

vodml-id: CartesianPosition.y
type: coords:Y
multiplicity: 1
Y-Axis value.

7.9.3 CartesianPosition.z

vodml-id: CartesianPosition.z
type: coords:Z
multiplicity: 1
Z-Axis value.

8 Velocity

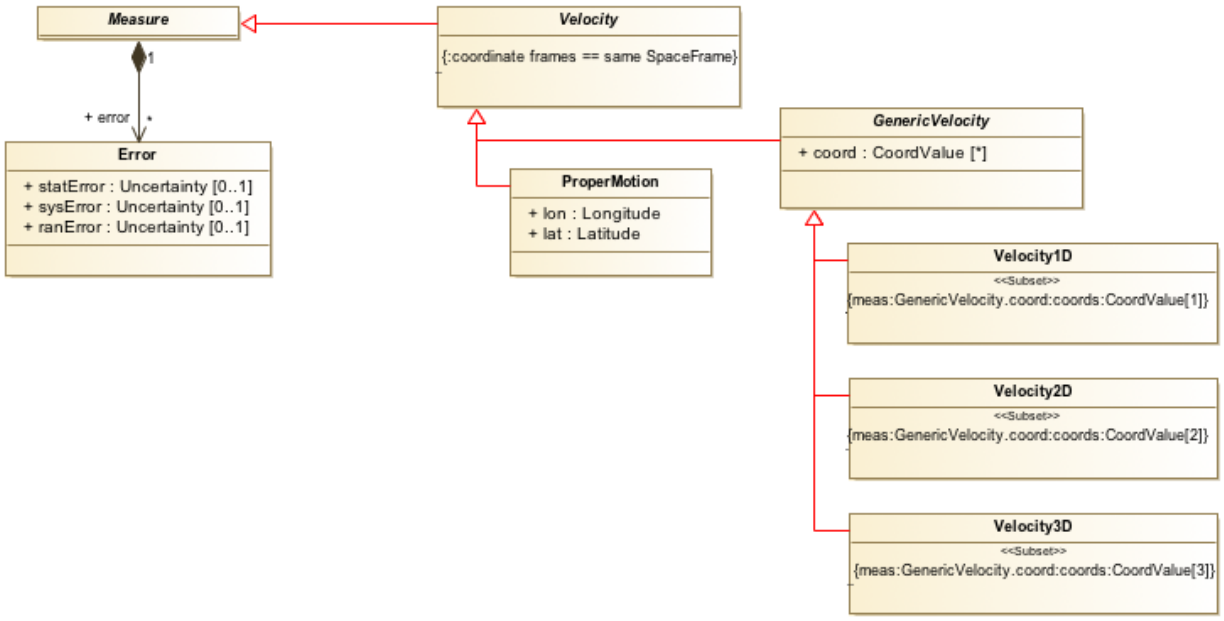

Figure 7: Velocity Measure elements

8.1 Velocity (Abstract)

Provides a comple description of a measured Velocity instant.
constraint
detail: Velocity.coordinate frames == same SpaceFrame

8.2 GenericVelocity (Abstract)

Generic velocity instant, appropriate for describing velocity data not covered by the specialized classes. Any concrete extension of this class MUST define a specific dimensionality for the coord attribute. We provide 1,2 and 3-dimensional classes to cover the most common cases.

8.2.1 GenericVelocity.coord

vodml-id: GenericVelocity.coord
type: coords:CoordValue
multiplicity: 0..*
The measured velocity value. May be given by any CoordValue type associated with a SpaceFrame.

8.3 Velocity1D

Generic 1-Dimensional velocity.

constraint
detail: Velocity1D.coords:CoordValue[1]

8.4 Velocity2D

Generic 2-Dimensional velocity.
constraint
detail: Velocity2D.coords:CoordValue[2]

8.5 Velocity3D

Generic 3-Dimensional velocity.
constraint
detail: Velocity3D.coords:CoordValue[3]

8.6 ProperMotion

Proper motion represented as the velocity in Longitude and Latitude directions of a spherical coordinate space. The associated SpaceFrame provides the details regarding the nature of the coordinate space (eg: Equatorial, Galactic, etc).

8.6.1 ProperMotion.lon

vodml-id: ProperMotion.lon
type: coords:Longitude
multiplicity: 1
Velocity in angular distance per unit time along the Longitude axis.

8.6.2 ProperMotion.lat

vodml-id: ProperMotion.lat
type: coords:Latitude
multiplicity: 1
Velocity in angular distance per unit time along the Latitude axis.

9 Polarization

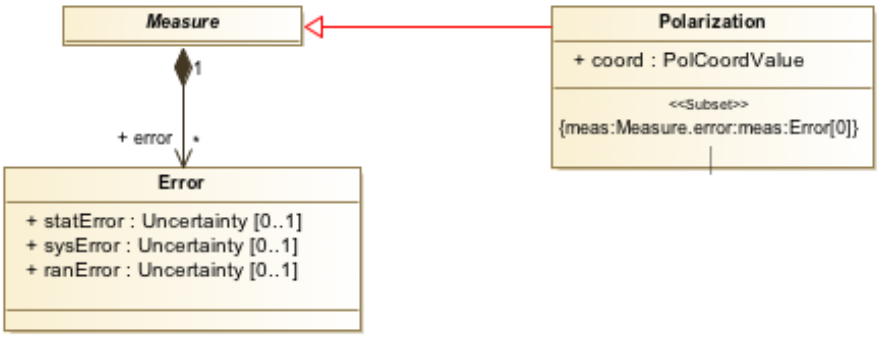

Figure 8: Polarization Measure elements

9.1 Polarization

Provides a complete description of a determined polarization state. Since the polarization coordinate is an enumerated type, there can be no associated numerical error sources.
constraint
detail: Polarization.meas:Error[0]

9.1.1 Polarization.coord

vodml-id: Polarization.coord
type: coords:PolCoordValue
multiplicity: 1
Determined polarization state. May be provided by any of the PolCoordValue subtypes.

10 Uncertainty

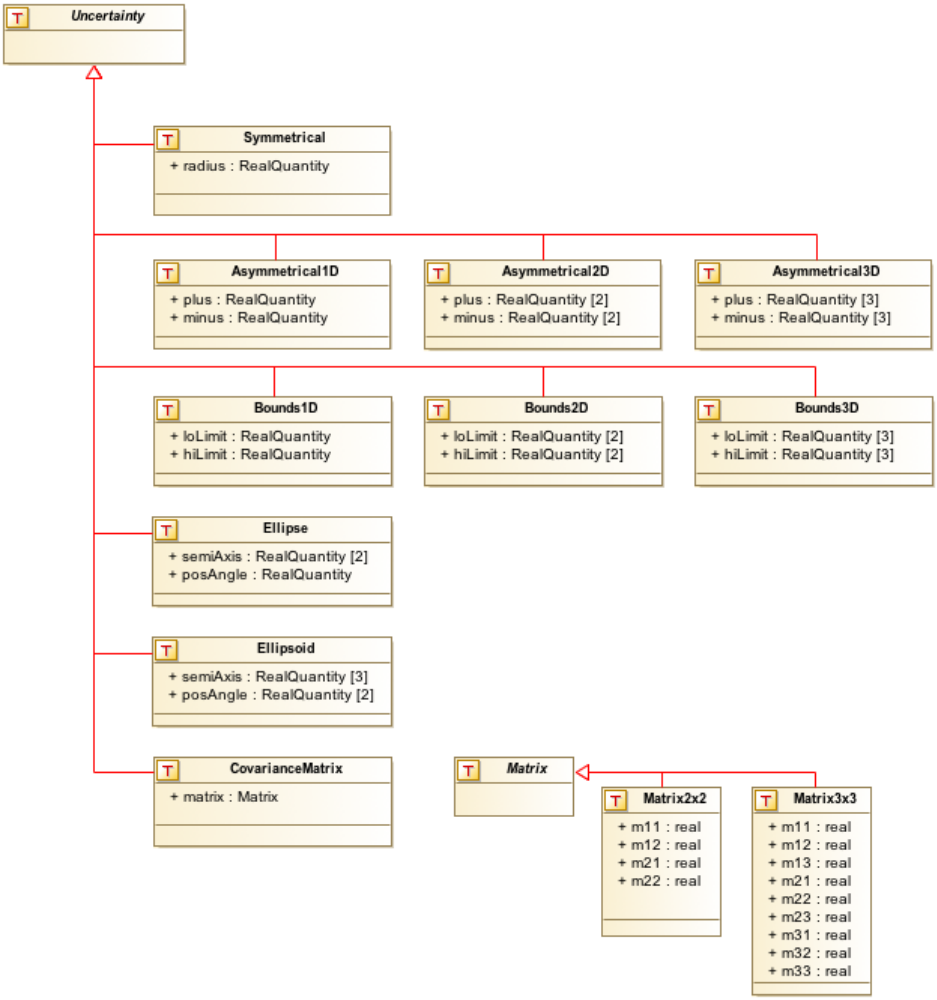

Figure 9: Uncertainty elements

10.1 Uncertainty (Abstract)

Abstract head of uncertainty types. These classes define the shape of the uncertainty, and are designed to be reusable in different contexts. Uncertainties are designed to be associated with a Coordinate or other object which provides the 'center' or reference location about which the uncertainty resides. In this model, we use them in the context of defining measurement errors, but they are also compatible for use in defining resolutions which are to be modeled at a later date.

10.2 Symmetrical

Symmetrical uncertainty, constant in all dimensions and directions from the associated Coordinate. ie: PlusMinus in 1D, circular in 2D, spherical in 3D.

10.2.1 Symmetrical.radius

vodml-id: Symmetrical.radius
type: ivoa:RealQuantity
multiplicity: 1
The uncertainty extent, constant in all dimensions and directions.

10.3 Asymmetrical1D

Uncertainty with different extent in the positive and negative directions from the associated Coordinate.

10.3.1 Asymmetrical1D.plus

vodml-id: Asymmetrical1D.plus
type: ivoa:RealQuantity
multiplicity: 1
Extent in the positive axis direction.

10.3.2 Asymmetrical1D.minus

vodml-id: Asymmetrical1D.minus
type: ivoa:RealQuantity
multiplicity: 1
Extent in the negative axis direction.

10.4 Asymmetrical2D

2D Uncertainty with different extent in the positive and negative axis directions from the associated Coordinate. i.e.: an offset rectangle.

10.4.1 Asymmetrical2D.plus

vodml-id: Asymmetrical2D.plus
type: ivoa:RealQuantity
multiplicity: 2
Extent in each positive axis direction.

10.4.2 Asymmetrical2D.minus

vodml-id: Asymmetrical2D.minus
type: ivoa:RealQuantity
multiplicity: 2
Extent in each negative axis direction.

10.5 Asymmetrical3D

3D Uncertainty with different extent in the positive and negative axis directions from the associated Coordinate. i.e.: an offset box.

10.5.1 Asymmetrical3D.plus

vodml-id: Asymmetrical3D.plus
type: ivoa:RealQuantity
multiplicity: 3
Extent in each positive axis direction.

10.5.2 Asymmetrical3D.minus

vodml-id: Asymmetrical3D.minus
type: ivoa:RealQuantity
multiplicity: 3
Extent in each negative axis direction.

10.6 Bounds1D

Provide the edges of the uncertainty space. Rather than being relative to the associated Coordinate, these represent a range within that Coordinate space.

10.6.1 Bounds1D.loLimit

vodml-id: Bounds1D.loLimit
type: ivoa:RealQuantity
multiplicity: 1
Lower limit of the uncertainty range.

10.6.2 Bounds1D.hiLimit

vodml-id: Bounds1D.hiLimit
type: ivoa:RealQuantity
multiplicity: 1
Upper limit of the uncertainty range.

10.7 Bounds2D

Provide the edges of a 2D uncertainty space. Rather than being relative to the associated Coordinate, these represent ranges along each axis of that Coordinate space.

10.7.1 Bounds2D.loLimit

vodml-id: Bounds2D.loLimit
type: ivoa:RealQuantity
multiplicity: 2
Lower edges of the uncertainty rectangle.

10.7.2 Bounds2D.hiLimit

vodml-id: Bounds2D.hiLimit
type: ivoa:RealQuantity
multiplicity: 2
Upper edges of the uncertainty rectangle.

10.8 Bounds3D

Provide the edges of a 3D uncertainty space. Rather than being relative to the associated Coordinate, these represent ranges along each axis of that Coordinate space.

10.8.1 Bounds3D.loLimit

vodml-id: Bounds3D.loLimit
type: ivoa:RealQuantity
multiplicity: 3
Lower edges of the uncertainty box.

10.8.2 Bounds3D.hiLimit

vodml-id: Bounds3D.hiLimit
type: ivoa:RealQuantity
multiplicity: 3
Upper edges of the uncertainty box.

10.9 Ellipse

Elliptical uncertainty shape.

10.9.1 Ellipse.semiAxis

vodml-id: Ellipse.semiAxis
type: ivoa:RealQuantity
multiplicity: 2
Extent of the semi-major and semi-minor axes, provided in the order of the associated Coordinate axes.

10.9.2 Ellipse.posAngle

vodml-id: Ellipse.posAngle
type: ivoa:RealQuantity
multiplicity: 1
Position angle, counter-clockwise from the positive direction of the first axis of the associated Coordinate.

10.10 Ellipsoid

Ellipsoidal uncertainty shape.

10.10.1 Ellipsoid.semiAxis

vodml-id: Ellipsoid.semiAxis
type: ivoa:RealQuantity
multiplicity: 3
Extent of the semi axes, provided in the order of the associated Coordinate axes.

10.10.2 Ellipsoid.posAngle

vodml-id: Ellipsoid.posAngle
type: ivoa:RealQuantity
multiplicity: 2
Position angles, counter-clockwise from the positive direction of the first axis toward the second axis, and angle 'above' the plane of the first two axes of the associated Coordinate.

10.11 CovarianceMatrix

Uncertainty expressed as a covariance matrix.

10.11.1 CovarianceMatrix.matrix

vodml-id: CovarianceMatrix.matrix
type: meas:Matrix
multiplicity: 1
The m x m matrix, where m is the number of dimensions of the associated Coordinate.

10.12 Matrix (Abstract)

Abstract Matrix data type.

10.13 Matrix2x2

2x2 matrix data type.

10.13.1 Matrix2x2.m11

vodml-id: Matrix2x2.m11
type: ivoa:real
multiplicity: 1
matrix cell; i=1, j=1

10.13.2 Matrix2x2.m12

vodml-id: Matrix2x2.m12
type: ivoa:real
multiplicity: 1
matrix cell; i=1, j=2

10.13.3 Matrix2x2.m21

vodml-id: Matrix2x2.m21
type: ivoa:real
multiplicity: 1
matrix cell; i=2, j=1

10.13.4 Matrix2x2.m22

vodml-id: Matrix2x2.m22
type: ivoa:real
multiplicity: 1
matrix cell; i=2, j=2

10.14 Matrix3x3

3x3 matrix data type.

10.14.1 Matrix3x3.m11

vodml-id: Matrix3x3.m11
type: ivoa:real
multiplicity: 1
matrix cell; i=1, j=1

10.14.2 Matrix3x3.m12

vodml-id: Matrix3x3.m12
type: ivoa:real
multiplicity: 1
matrix cell; i=1, j=2

10.14.3 Matrix3x3.m13

vodml-id: Matrix3x3.m13
type: ivoa:real
multiplicity: 1
matrix cell; i=1, j=3

10.14.4 Matrix3x3.m21

vodml-id: Matrix3x3.m21
type: ivoa:real
multiplicity: 1
matrix cell; i=2, j=1

10.14.5 Matrix3x3.m22

vodml-id: Matrix3x3.m22
type: ivoa:real
multiplicity: 1
matrix cell; i=2, j=2

10.14.6 Matrix3x3.m23

vodml-id: Matrix3x3.m23
type: ivoa:real
multiplicity: 1
matrix cell; i=2, j=3

10.14.7 Matrix3x3.m31

vodml-id: Matrix3x3.m31
type: ivoa:real
multiplicity: 1
matrix cell; i=3, j=1

10.14.8 Matrix3x3.m32

vodml-id: Matrix3x3.m32
type: ivoa:real
multiplicity: 1
matrix cell; i=3, j=2

10.14.9 Matrix3x3.m33

vodml-id: Matrix3x3.m33
type: ivoa:real
multiplicity: 1
matrix cell; i=3, j=3

A Changes from Previous Versions

No previous versions yet.

B Modeling Conventions

This model follows the VO-DML modeling practices, however, the UML representations may vary depending on the tool used. Below we describe the graphical representation of the modeling concepts and relations.

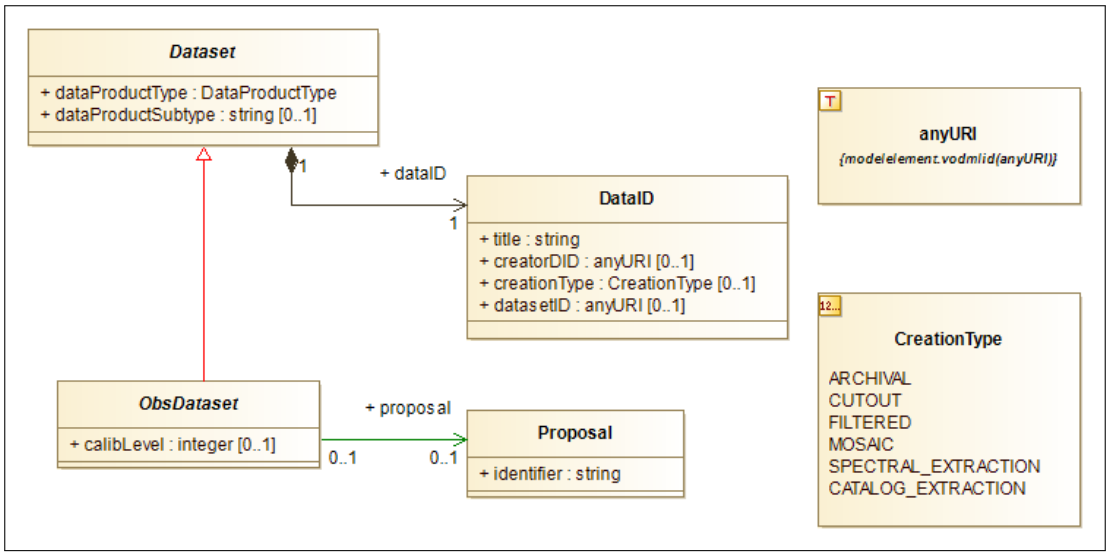

Figure 10: Notation example diagram

B.1 Class

Classes are represented by a plain box. The class name is annotated in the top window, abstract classes use italic typeface. Attributes, if any, are listed in the lower panel. Attributes may only be of primitive type (real, string, etc), a defined DataType, or an Enumeration type. Relationships to other objects are defined via the composition and reference relation arrows.

B.2 DataType

DataTypes are represented by a box shape similar to Class, but annotated with a "T" symbol in the top left corner.

B.3 Enumerations

Enumerations are represented by a box shape similar to Class, but annotated with a "1,2.." symbol in the top left corner. Enumeration Literals (possible values) are listed below the enumeration class name.

B.4 Generalization

Generalizations are represented by a red line, with open triangle at the end of the source, or more general, object.

B.5 Composition

The composition relation is indicated by a black line with a solid diamond attached to the containing object, and an arrow pointing to the object being contained. The composition relation is very tight, where the container is responsible for the creation and existence of the target. Any object may be in no more than one composition relation with any container. The attribute name for the composition relation is annotated at the destination of the relation (e.g. "+ dataID"). This is typically a lower-cased version of the destination class name, but this is not required.

B.6 Reference

The reference relation is indicated by a green line, with an arrow pointing to the object being referenced. The reference relation is much looser than composition, the container has no ownership of the target, but merely holds a pointer, or other indirect connection to it. The attribute name is annotated at the destination of the relation (e.g. "+ proposal"). This is typically a lower-cased version of the destination class name, but may be another name indicating the role that the class is playing in this context.

B.7 Multiplicity

All attributes and relations have a multiplicity associated with them. For attributes, the multiplicity is contained within brackets just after the attribute name. If no bracket is displayed, this is equivalent to '[1]'.

- 1 = one and only one value must be provided.
- 0..1 = zero or one value may be provided.
- * = zero or more values may be provided (open ended).

C Data Types

C.1 Base Data Types

Provides a set of standardized primitive data types as well as types for representing quantities (values with associated units). We provide a diagram of the model here, and refer the reader to Section 5 of the VO-DML modeling specification document (Lemson and Laurino et al., 2018) for more information.

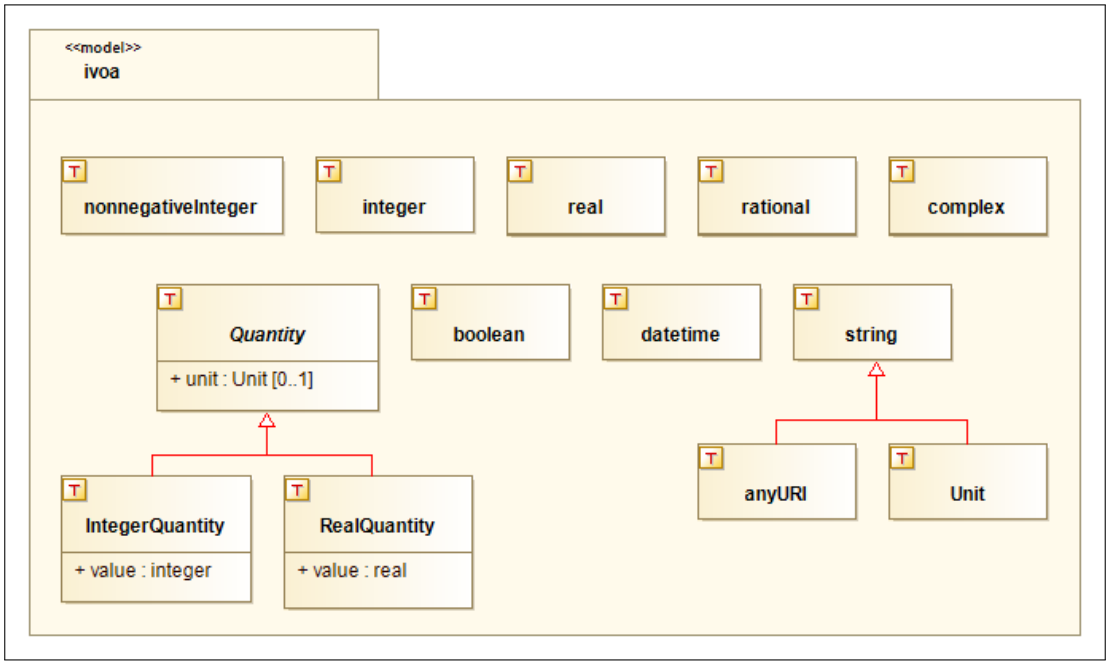

Figure 11: Base Data Types

C.1.1 Units

This model requires the use of the IVOA VOUnits Standard (Demleitner and Derriere et al., 2014) for representing units of physical quantities. This standard reconciles common practices and current standards for use within the IVOA community.

C.1.2 Dates

The 'datetime' datatype is for expressing date-time values. The string representation of a date-time value should follow the FITS convention for representing dates. The FITS standard is effectively ISO8601 format without the "Z" tag to indicate UTC (YYYY-MM-DDThh:mm:ss). Values are nominally expressed in UTC.

References

Arviset, C., Gaudet, S. and the IVOA Technical Coordination Group (2010), 'IVOA architecture', IVOA Note.
http://www.ivoa.net/documents/Notes/IVOAArchitecture

Bradner, S. (1997), 'Key words for use in RFCs to indicate requirement levels', RFC 2119.
http://www.ietf.org/rfc/rfc2119.txt

Demleitner, M., Derriere, S., Gray, N., Louys, M. and Ochsenbein, F. (2014), 'Units in the VO, version 1.0', IVOA Recommendation.
http://www.ivoa.net/documents/VOUnits/index.html

Lemson, G., Laurino, O., Bourges, L., Cresitello-Dittmar, M., Demleitner, M., Donaldson, T., Dowler, P., Graham, M., Gray, N., Michel, L. and Salgado, J. (2018), 'Vodml: a consistent modeling language for ivoa data models, version 1.0', IVOA Recommendation.
http://www.ivoa.net/documents/VODML/index.html

Rots, A. (2007), 'Space-time coordinate metadata for the virtual observatory', IVOA Recommendation.
http://www.ivoa.net/documents/latest/STC.html

Rots, A., Cresitello-Dittmar, M. and Laurino, O. (2019), 'Astronomical coordinates and coordinate systems, version 1.0', IVOA Working Draft.
http://www.ivoa.net/documents/Coords/