International
Virtual
Observatory
Alliance

IVOA Dataset Metadata Model

Version 1.0

IVOA Working Draft 20151007

This version:

WD-DatasetDM-1.0-20151007

Previous version(s):

Editor(s):

Mark Cresitello-Dittmar

Authors:

Francois Bonnarel, Omar Laurino, Gerard Lemson, Mireille Louys, Arnold Rots, Doug Tody, and the IVOA Data Model Working Group.

Abstract

This document provides a data model describing the structure and content of generic Dataset metadata for the IVOA. This is a high-level model which is to be referenced and extended by other models describing specific types of Datasets and Data products. In this document, we specify the generic Dataset, as well as an ObservationDataset model which covers the class of Datasets which are derived from an Observation. At the time of this writing, there is no formal Observation-Experiment model for the IVOA, so we include a hypothetical Observation-Experiment model to serve as a placeholder.

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".

Acknowledgements

This document has been developed with support from NSF and NASA under the Virtual Astronomical Observatory (VAO) project, the National Science Foundation's http://www.nsf.gov Information Technology Research Program under Cooperative Agreement AST0122449 with The Johns Hopkins University, from the UK Particle Physics and Astronomy Research Council (PPARC) http://www.pparc.ac.uk, and from the Euro-VO projects (European Commission 7th program): Euro-VO Aida, VO-ICE and CoSADIE.

Change Log:

2014 Jun 13: Initial draft of version 1.0
2014 Sep 30: Draft revised with STC2 prototype, initial draft feedback, and updates due to Cube model development.
2015 Mar 02:
- Incorporate comments from Spectral2.0 feedback related to Dataset metadata.
- Format change to better illustrate data type and multiplicity of elements.
- Update STC2 prototype to current state of development.
- General review of element descriptions for clarity.

2015 Oct 07:
- Update Acknowledgments for European contributors
- Generalize Derived object using ObsConfig pattern
- Remove Redshift type (used in Derived)
- Update STC2 prototype to 2015-05-04 model descriptions (except for Transform model)
- Update reference to VO-DML specification (2015-02-06 version)
- Make section references links.

Contents

1 Introduction

1.1 Motivation

All IVOA datasets must contain a common set of metadata elements to facilitate the registration, discovery, and interoperability of these datasets. To date, individual IVOA data models have independently defined this metadata within the separate documents. This has resulted in some level of inconsistency between models, as well as document bloat, and some ambiguity as to the hierarchy and relation of models to each other. For example, the ObsCore-1.0 model describes itself as defining "the core components of the Observation data model ", but there is no formal definition of an Observation data model in the IVOA. Without this higher-level document, it is difficult for detailed models to properly reference and/or extend this content consistently.

With the development of the Cube model, significant effort has been made to properly model this high-level metadata, and separate the components related to the generic dataset, a dataset derived from an observation, and the observation itself. This document represents the results of that effort. Here, we define the generic dataset metadata, and provide an example for extending this with metadata related to datasets resulting from a specific process (Observation). As such, the ObsCore model should be considered a 'view' of this model, highlighting the core components required for supporting TAP services.

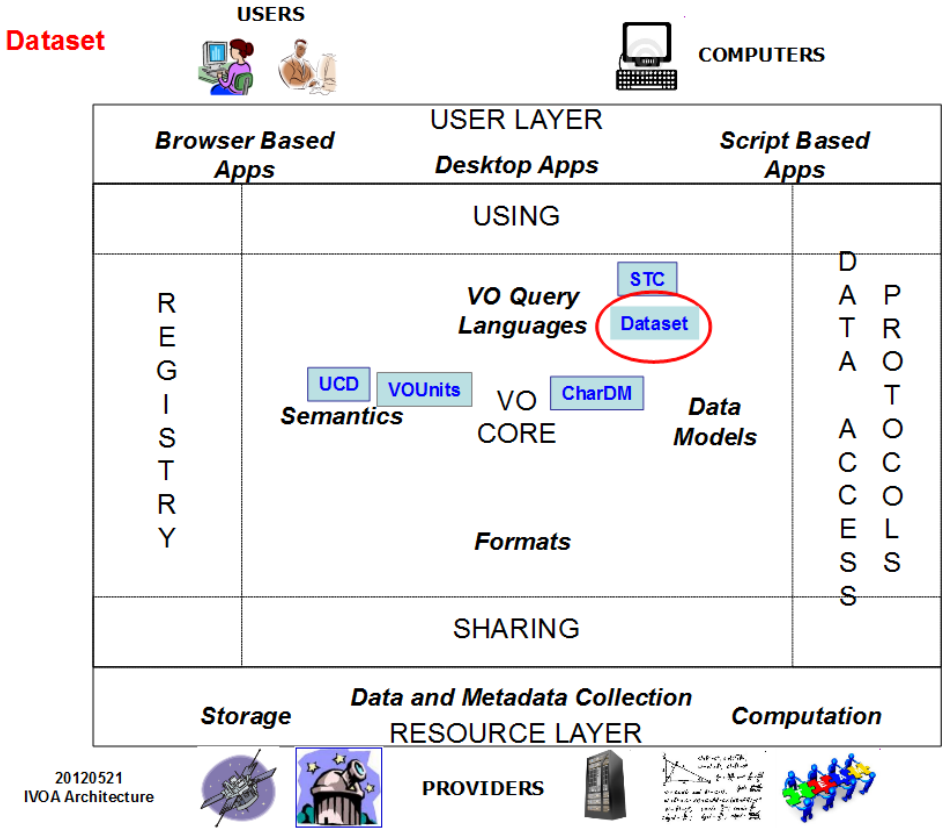
The descriptions of many elements of this model are a result of reviewing and combining those contained in the ObsCore (1.0), Spectral (2.0), and Characterisation (1.13) models. As such, it represents a uniform, consistent description set. Future revisions of those documents should be defined with respect to this model.

1.2 Requirements

The primary goals of this document are:

- to provide a specification of generic dataset metadata.
- to specify metadata associated with an Observation (experiment) which are to be included in datasets derived from observations (ObsDataset).

1.3 Role in the IVOA Architecture



1.4 Model Dependencies

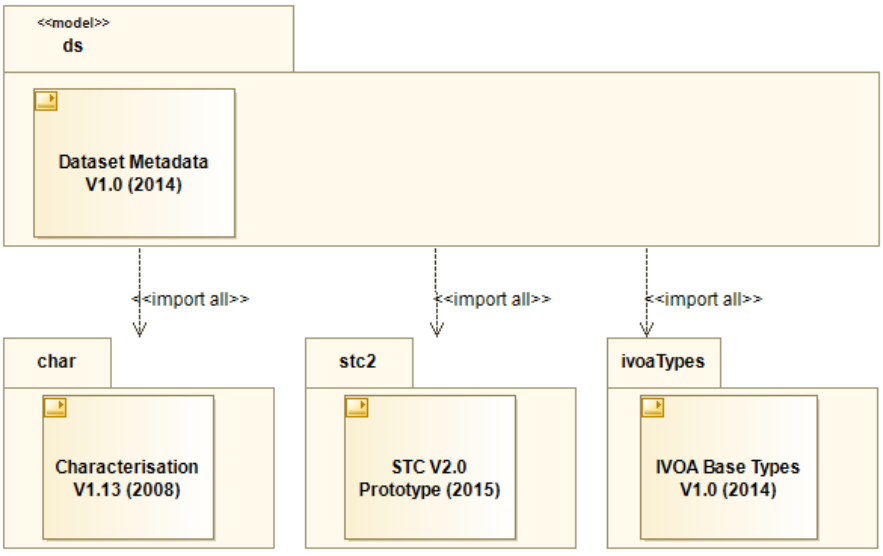


Figure 1: Model dependency diagram

The Dataset model is built on other data models as indicated in Figure 1. The <<model>> and <<import>> stereotypes provide information identifying the model, it's version, any dependencies, and URLs to find more information about the model definitions including HTML and schema documentation. See Appendix B for more information about the content of these stereotypes and how they are used in serializations.

1.5 Structure of this Documentation

+ Major sections for each model area (Dataset, Observation, etc.).
+ First subsection in each section is the primary element within that model
+ Subsequent subsections for secondary elements, generally in alphabetical order, but occasionally a logical grouping of related objects makes more sense.
+ Each subsection has sub-subsections for each attribute/relation
    - attributes show the full definition including datatype and usage.
    - relations describe the usage of the object in that context, the type of the target of the relation, and a reference to the full definition of that type.
+ Capitalization convention
    - Objects and complex data types are expressed in PascalCase
    - Attributes are camelCase
    - Primitive data types (string, double, etc.) are lower case

2 Dataset Model

This section describes the generic, high-level metadata associated with an IVOA Dataset. Since serialization format choices may effect the number of files or components which comprise a dataset, we define an IVOA Dataset as "a file or files which are considered to be a single deliverable". Examples of viable datasets include:

+ An individual data product, such as a Spectrum, or Image.

+ A 'tar' file or directory of processed observational data files.

This metadata identifies the dataset, and provides information regarding the ownership, rights and associations with other datasets. The primary purpose of this metadata is to facilitate the registry and discovery of datasets within the IVOA community.

Several of the objects modeled here are based on descriptions given in the IVOA document, "Resource Metadata for the Virtual Observatory; Version 1.12" [1] (Resource Metadata). Where applicable, we provide the appropriate citation in the text below.

2.1 Dataset

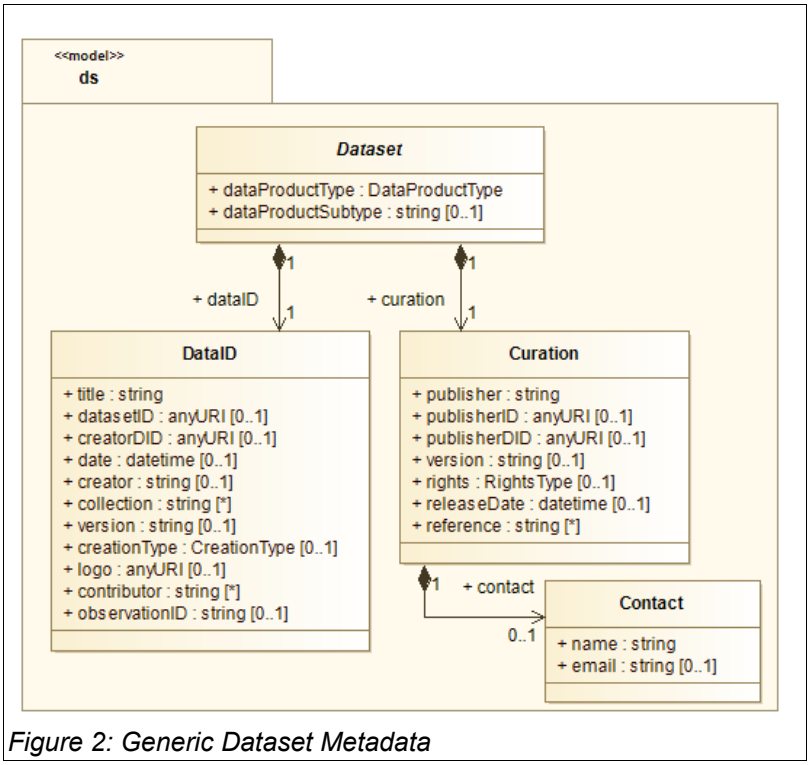


Figure 2: Generic Dataset Metadata

Abstract object for the generic IVOA Dataset. It is intended to be useful for any type of data. Specific dataset models should extend this object, providing detailed definitions and additional content as appropriate for that type of dataset.

2.1.1 Dataset.dataProductType

type: DataProductType **type-detail: Section 5.2.2**
multiplicity: 1

Describes the high level scientific classification of the data content. Values are restricted to the DataProductType enumeration set and convey the general idea of the content and organization of a dataset.

2.1.2 Dataset.dataProductSubtype

type: string **type-detail: Section 5.1**
multiplicity: 0..1

Secondary type classification for the dataset. This field is intended to precisely specify the scientific nature of the data product, possibly in terms relevant only to a specific archive or data collection. For example, dataProductType='image' could have associated dataProductSubtype="src.image", "bkg.image", "PixelMask", etc. Values are unrestricted strings.

2.1.3 Dataset.curation

type: Curation **type-detail: Section 2.3**
multiplicity: 1

Provides metadata related to the entity responsible for the curation of the dataset.

2.1.4 Dataset.dataID

type: DataID **type-detail: Section 2.4**
multiplicity: 1

DataID provides high level identification metadata for the dataset itself, and any associations with various collections.

2.2 Contact

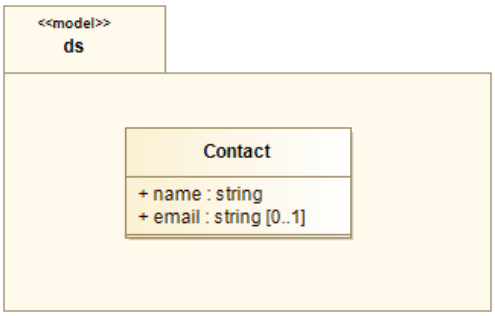


Figure 3: Contact detail diagram

Contact information for a person or entity.

2.2.1 Contact.name

type: string **type-detail: Section 5.1**
multiplicity: 1

Name of the contact or entity. (RM:Contact.Name)

2.2.2 Contact.email

type: string **type-detail: Section 5.1**
multiplicity: 0..1

E-mail address of the contact. (RM:Contact.Email)

2.3 Curation

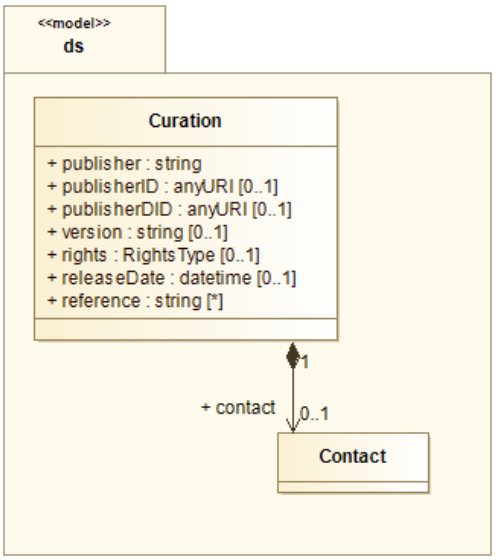


Figure 4: Curation detail diagram

The Curation object provides metadata about the entity responsible for the support of the dataset content. It is assembled from definitions provided by the IVOA Resource Metadata document. Here, we provide a brief description of each field for easy reference, along with a notation of its mapping to the Resource Metadata document (RM:field), where the reader may find more detailed information.

2.3.1 Curation.publisher

type: string **type-detail: Section 5.1**
multiplicity: 1

Common name for the entity making the data available. (RM:Curation.Publisher)

2.3.2 Curation.publisherID

type: anyURI **type-detail: Section 5.1**
multiplicity: 0..1

IVOA resource identifier associated with the publisher and registered with an IVOA compliant registry (eg: ivo://mast.stsci.edu). Values are to be expressed using the syntax described in "IVOA Identifiers"[2]. (RM:Curation.PublisherID)

2.3.3 Curation.publisherDID

type: anyURI **type-detail: Section 5.1**
multiplicity: 0..1

IVOA dataset identifier assigned by the publisher to uniquely identify the dataset within its holdings. Typically, the basis of this identifier will be the publisher ID. However, if the publisher chooses to use a 'global index service' such as ADS to obtain persistent identifiers for their

datasets, rather than generate their own, that identifier should be used both here and for DataID.datasetID. Note: this model also defines a creator dataset ID (DataID.creatorDID), these will differ if the publishing entity is not the creator of the dataset. Values are to be expressed as dataset identifiers using the syntax described in "IVOA Identifiers"[2].

2.3.4 Curation.releaseDate

type: datetime **type-detail: Section 5.1.3**
multiplicity: 0..1

Date the curated dataset was last modified. (RM:Curation.Date)

2.3.5 Curation.version

type: string **type-detail: Section 5.1**
multiplicity: 0..1

Version of the curated dataset, assigned by the publisher. This is an independent versioning from DataID.version that allows the publisher to track changes to the high level dataset metadata (e.g. curation metadata, identifiers, etc.) without effecting the creator defined dataset version. The value may be based on the DataID.version (e.g. by adding a sub-version extension), or an independent versioning. There are no format restrictions on the value. (RM:Curation.Version)

2.3.6 Curation.rights

type: RightsType **type-detail: Section 5.2.3**
multiplicity: 0..1

Indicates the access privileges to the content. Values are restricted to the RightsType enumeration set. (RM:Collection.Rights)

2.3.7 Curation.reference

type: string **type-detail: Section 5.1**
multiplicity: 0..*

Zero or more bibliographic or documentation references associated with the dataset. Each instance provides a single forward link to a major publication which references the dataset. Values should be expressed as a URL, or bibcode (discernible as a 19 character string beginning with 4 digits). Free text references are allowed, but discouraged. (RM:General.Source)

2.3.8 Curation.contact

type: Contact **type-detail: Section 2.2**
multiplicity: 0..1

Contact information of the person/entity responsible for the content of the dataset. We recommend using a generic 'helpdesk' type contact rather than individuals whose information may more easily become obsolete. (RM:Curation.Contact)

2.4 DataID

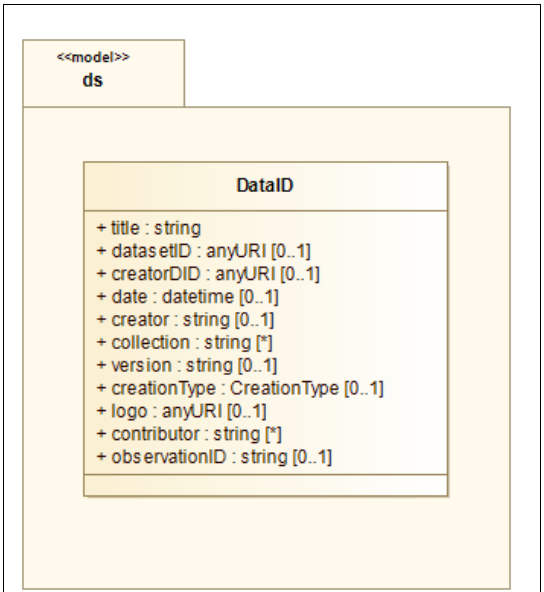


Figure 5: DataID detail diagram

The Data Identification object (DataID) stores the dataset identifiers and its membership within larger collections.

The Dataset IDs in this object must comply with the syntax for dataset identifiers defined in the "IVOA Identifiers" [2] document, including the use of 'stop' characters to identify specific datasets that are not individually in the registry. e.g., ivo://example.net/aservice?2013/5/2342.

Much of the content of this object is assembled from various definitions in the IVOA Resource Metadata document. Here, we provide a brief description of each field for easy reference, along with a notation of its mapping to the Resource Metadata document (RM:field), where the reader may find more detailed information.

2.4.1 DataID.title

type: string **type-detail: Section 5.1**
multiplicity: 1

A free form string giving a title for the dataset. (RM:Identity.Title)

2.4.2 DataID.datasetID

type: anyURI **type-detail: Section 5.1**
multiplicity: 0..1

If the dataset is registered with an external 'global index service' such as ADS, the publisher may include that identifier here. This provides a common, persistent identifier for the dataset, and possible access point to follow for information on publications and other related datasets. Note: the same dataset published at more than one location would have different Curation.publisherDID values, but the same DataID.datasetID. eg: "ivo://ADS/Sa.CXO?obsid=1234", "ivo://ADS/sh.hut#ngc4151_141"

2.4.3 DataID.creatorDID

type: anyURI **type-detail: Section 5.1**
multiplicity: 0..1

The dataset identifier assigned by the creator. Here, the authority-id of the identifier must be that of the creator. It is used to identify the original exposure of the dataset in an archive, and will remain static regardless of where the dataset is published. The creator ID will not necessarily change even if the VO object in question is a cutout or is otherwise further processed.

2.4.4 DataID.date

type: datetime **type-detail: Section 5.1.3**
multiplicity: 0..1

Data processing or creation date (RM:Curation.Date). See Section 5.1.3 for Date format specification.

2.4.5 DataID.creator

type: string **type-detail: Section 5.1**
multiplicity: 0..1

A free form string giving the name of the institution or entity which created the dataset. (RM:Curation.Creator)

2.4.6 DataID.collection

type: string **type-detail: Section 5.1**
multiplicity: 0..*

The dataset is associated with zero or more Collections (instrument name, survey name, etc.) . Each instance is a free form string of a particular collection tag. The values are generally defined by the creating entity and indicate some degree of compatibility with other data sharing the same Collection properties. Examples: "WFC", "Sloan", "BFS Spectrograph", "MSX Galactic Plane Survey".

2.4.7 DataID.version

type: string **type-detail: Section 5.1**
multiplicity: 0..1

Version assigned by the creator, reflecting the production version of the dataset. This value should only be changed by the creator, upon the new release of a dataset. There are no format restrictions or specifications on the versioning scheme.

2.4.8 DataID.creationType

type: CreationType **type-detail: Section 5.2.1**
multiplicity: 0..1

The dataset creation type describes the nature or genre of the content. Values are restricted to the CreationType enumeration set. (RM:General.Type).

2.4.9 DataID.logo

type: anyURI **type-detail: Section 5.1**
multiplicity: 0..1

URL pointer to a graphical logo associated with the creator of the document content. (RM:Curation.Creator.Logo)

2.4.10 DataID.contributor

type: string **type-detail: Section 5.1**
multiplicity: 0..*

Persons or entities who contributed to the generation of the scientific content of the dataset. Users of the dataset should include these in subsequent credits and acknowledgements. Each instance identifies an entity, formatted to the desired acknowledgment expression. (RM:Curation.Contributor)

2.4.11 DataID.observationID

type: string **type-detail: Section 5.1**
multiplicity: 0..1

Internal ID determined by the data provider to identify the observation from which the dataset was produced.

3 ObservationDataset (ObsDataset)

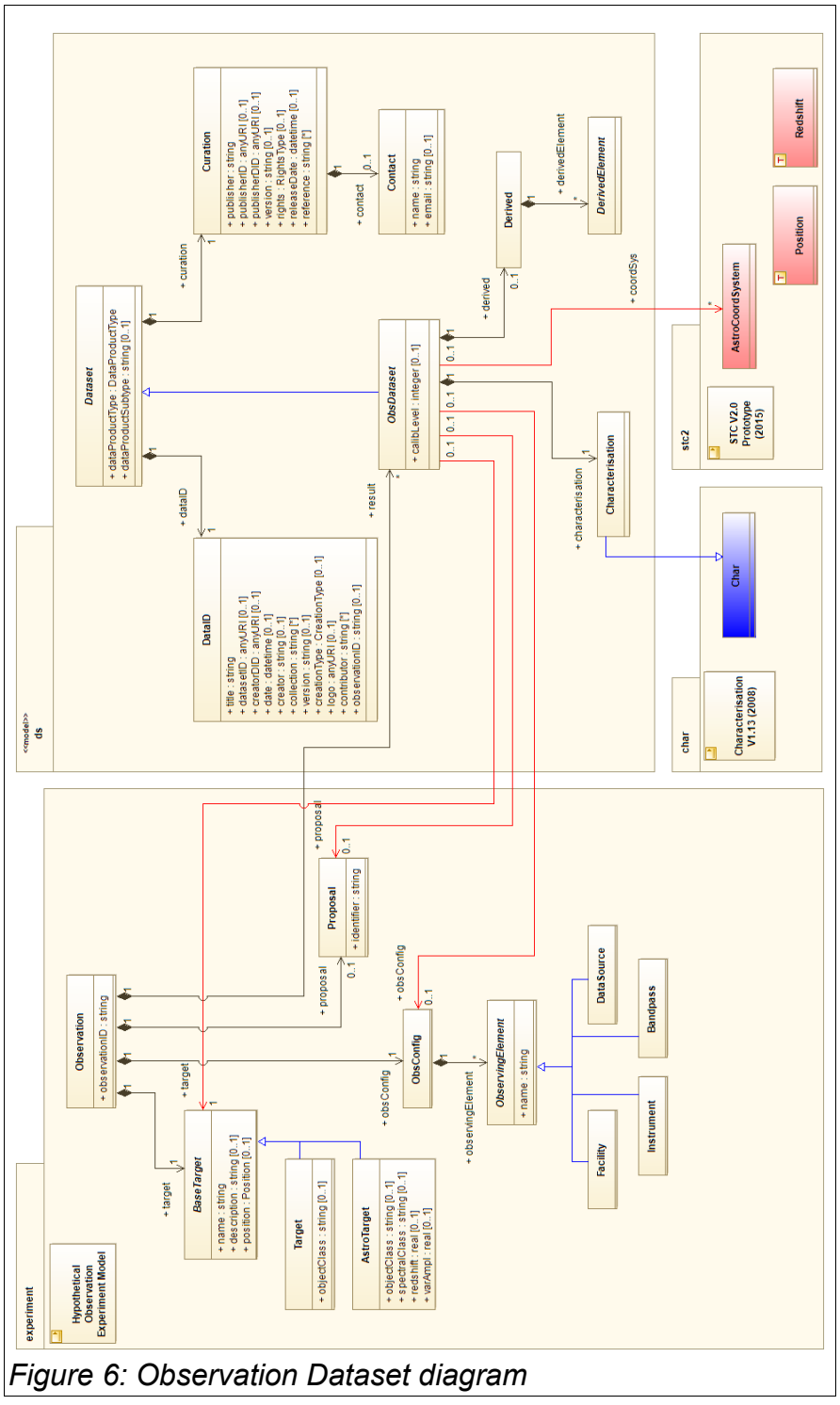


Figure 6: Observation Dataset diagram

This section defines additional metadata associated with Datasets which are derived from Observations. This metadata gives a high-level summary of the coverage of the dataset in coordinate space, as well as the coordinate systems used, and general information about the observation itself.

Many components of this object are expected to be sourced from an as yet undefined Observation model. In lieu of this model, we define a straw-man substitute in this document. The Observation model described here depicts an "Observation" as a type of "Experiment", where certain elements of the observation configuration and parameters are referenced within the ObsDataset. Other forms of "Experiment" (such as "Simulation") could fit in this framework. In fact, much of the framework is based on that shown in the Simulation data model[6].

3.1 ObsDataset

Abstract object extending Dataset with metadata relevant to datasets derived from Observations.

3.1.1 ObsDataset.calibLevel

type: integer **type-detail: Section 5.1**
multiplicity: 0..1

High level classification for the calibration level of a particular dataset as a whole. The calibration level concept conveys to the user information on how much data reduction/processing has been applied to the data. It is up to the data providers to consider how to map their own internal classification to the scale defined here.

Scale:

- 0 - Raw instrumental data, in a proprietary or internal data-provider defined format.
- 1 - Instrumental data in a standard format (FITS, VOTable, etc)
- 2 - Calibrated, science ready data with the instrument signature removed.
- 3 - Enhanced data products like mosaics, resampled or drizzled images, or heavily processed survey fields. Level 3 data products may represent the combination of data from multiple primary observations.

3.1.2 ObsDataset.characterisation

type: Characterisation **type-detail: Section 3.2**
multiplicity: 1

Characterisation provides a 'characteristic' view of the dataset coordinate space. For each represented domain (spatial, spectral, temporal, etc), characterisation provides metadata summarizing the coverage, resolution, representative accuracies, etc. for the dataset as a whole.

3.1.3 ObsDataset.coordSys

type: AstroCoordSystem **type-detail: Section 6.1.2**
multiplicity: 0..*

Zero or more references to coordinate system defintions associated with the dataset. Since multiple data products may contibute to the content of a dataset, this element provides convenient, high-level access to definitions which may be distributed among lower-level objects.

3.1.4 ObsDataset.derived

type: Derived **type-detail: Section 3.3**
multiplicity: 0..1

Provides a high level summary of certain properties of the dataset. Its primary purpose is to support high level filtering of datasets during data discovery.

3.1.5 ObsDataset.obsConfig

type: ObsConfig **type-detail: Section 4.5**
multiplicity: 0..1

Reference to ObsConfig object from Observation. This object provides some high-level metadata related to the observation configuration.

3.1.6 ObsDataset.proposal

type: Proposal **type-detail: Section 4.7**
multiplicity: 0..1

Reference to Proposal object from Observation. This object provides metadata identifying any proposal related to the observation which produced the dataset.

3.1.7 ObsDataset.target

type: BaseTarget **type-detail: Section 4.2**
multiplicity: 1

Reference to a BaseTarget object from Observation. Provides metadata describing the target of the observation.

3.2 Characterisation

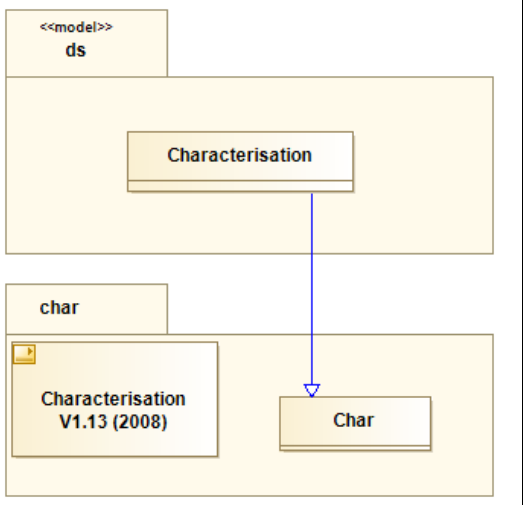


Figure 7: Characterisation Relation to Char Model

The ObsDataset Characterisation object is a direct extension of the Characterisation object from the IVOA Characterisation Data Model Standard [3]. This provides a local object which may be contained in a 'composition' relation within this model (specifically ObsDataset). This object may be extended and/or modified by specific Dataset models as needed.

Characterisation provides a broad scale description of the coordinate space occupied by the dataset in each of the axis domains. This includes axes not directly represented in the dataset 'data', for example, the Time axis of a 2D spatial image. As such, there is a strong correlation between Characterisation and Data objects. In some cases, the values may be directly derived from the data, in others, they provide supporting metadata about the broader domain. For example, the Accuracy fields in Characterisation represent typical accuracy for the dataset, while those within Data provide per-data-point errors.

3.3 Derived

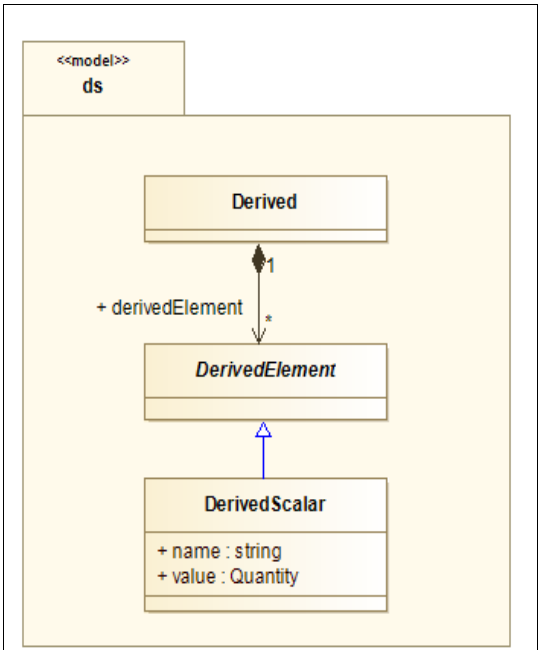


Figure 8: Derived detail diagram

The Derived (short for Derived Data) object holds derived information obtained by evaluating or analyzing the contents of the dataset. The specific content of this object is strongly dependent on the specific type of dataset, so we provide a generic model which may be specialized in other models to define elements appropriate for that type of dataset.

The primary purpose of this object is to provide a common framework in which specific information may be placed to aid in discovery and filtering of datasets in various access protocols.

3.3.1 Derived.derivedElement

type: DerivedElement **type-detail: Section 3.4**
multiplicity: 0..*

Collection of zero or more DerivedElement objects, each of which provides a specific quantity obtained by analyzing the dataset content.

3.4 DerivedElement

Abstract base for defining derived data elements. Typically, models for specific data products would extend this object to define various elements appropriate for that model. For example, the Spectrum model could define signal-to-noise ratio (SNR), or TimeSeries could define period, or variability. We put no restriction on the DerivedElement content since the result could be a simple value or a complex object. However, it is recommended that extensions be simple and compact in keeping with the primary intent of use in data discovery.

3.5 DerivedScalar

Simple extension of DerivedElement class which can serve many use cases. Usages of this object in other models to define specific elements should explicitly define the element name, and the process by which the value is determined.

3.5.1 DerivedScalar.name

type: string **type-detail: Section 5.1**
multiplicity: 1

Name identifying the derived element.

3.5.2 DerivedScalar.value

type: Quantity **type-detail: Section 5.1**
multiplicity: 1

Value of the derived element.

4 Observation-Experiment

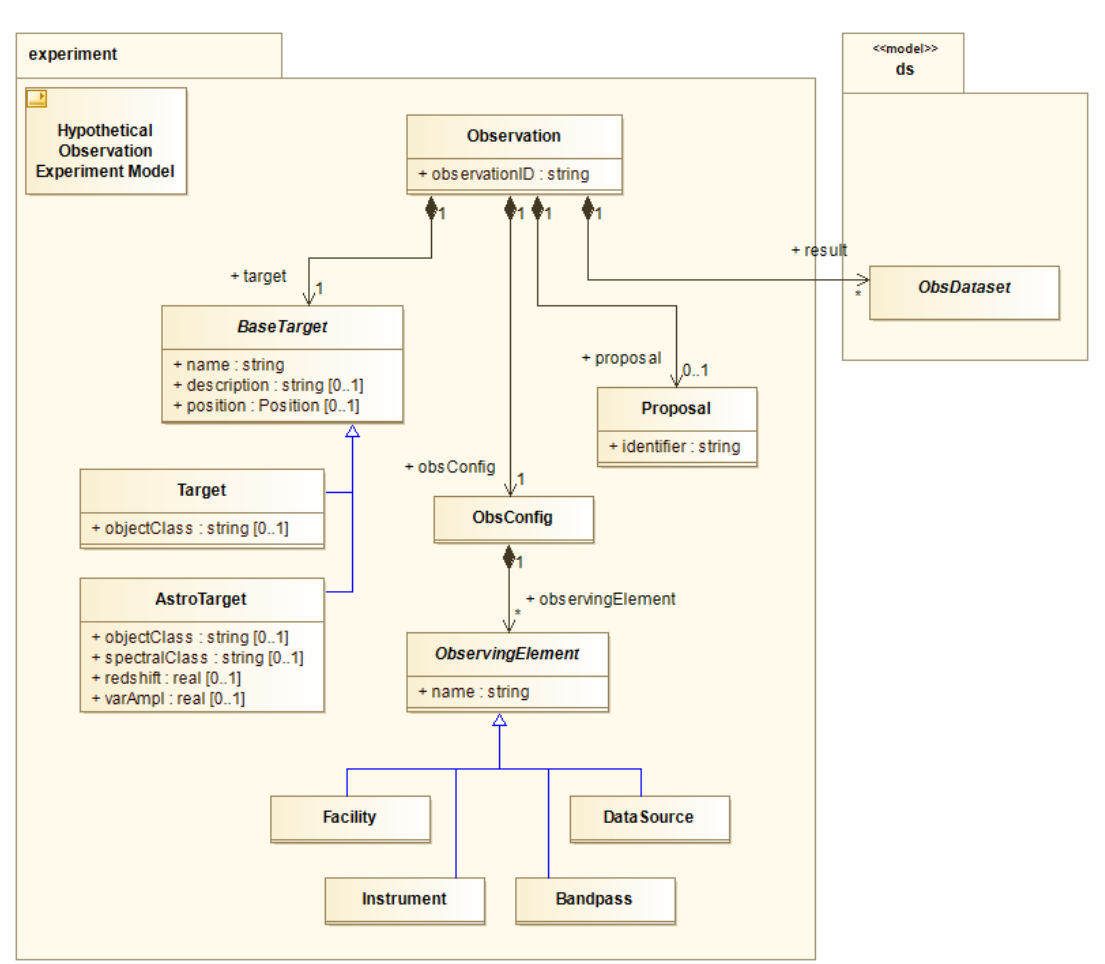


Figure 9: Observation Experiment detail diagram

The ObsDataset model refers to several elements related to an Observation and its configuration. As of the time of this writing, there is no IVOA recommendation for a general Observation data model. The Provenance data model, in progress, will define the pattern for describing the relation between actions and results, and how to record these in datasets. In lieu of these standards, this document defines a straw-man Observation model.

The Observation is modeled as a type of 'Experiment', with some basic structure defined to provide metadata about the observation target and configuration. The product, or 'result' of the Observation is zero or more ObsDataset objects. This pattern is inspired by, and compatible with the Simulation Data Model [6], where a 'Simulation' can be considered another form of 'Experiment' or perhaps even another form of 'Observation'.

4.1 Observation

4.1.1 Observation.observationID

type: string **type-detail: Section 5.1**
multiplicity: 1

Internal ID determined by the data provider to uniquely identify the observation within the institution or entity performing the observation.

4.1.2 Observation.target

type: BaseTarget **type-detail: Section 4.2**
multiplicity: 1

The target of the observation. The content of this object may vary greatly depending on the goals and nature of the observation. For example the 'target' could be a galaxy, stellar object, planet, or calibration source. As such, we allow the BaseTarget class here, and permit users to define and use more content rich flavors according to their needs.

4.1.3 Observation.obsConfig

type: ObsConfig **type-detail: Section 4.5**
multiplicity: 1

Observation configuration metadata, provides information about who, where, and how the observation was conducted.

4.1.4 Observation.proposal

type: Proposal **type-detail: Section 4.7**
multiplicity: 0..1

Identifies any proposal related to the observation. This field may be used to gather all observations and products related to a particular proposal.

4.1.5 Observation.result

type: ObsDataset **type-detail: Section 3.1**
multiplicity: 0..*

The result of an observation is zero or more Observation Datasets

4.2 BaseTarget

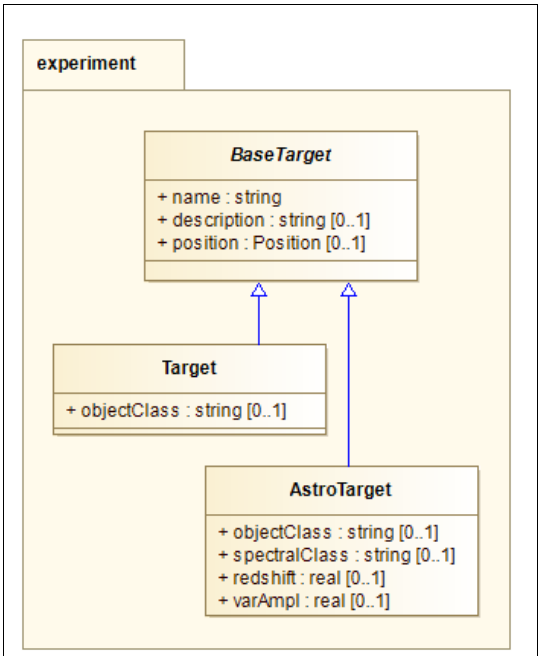


Figure 10: Target hierarchy detail diagram

Abstract base class for the Target object tree. The target object provides identifying metadata related to the subject or goal of the experiment. For an Observational experiment, this would typically be an astronomical object. The BaseTarget class defines high-level identifying information, and must be extended for particular classes of Target which may define additional characteristics.

4.2.1 BaseTarget.name

type: string **type-detail: Section 5.1**
multiplicity: 1

The target name. The primary purpose of this field is to provide the user with a recognizable identity of the particular subject or goal. However, since this may be a query-able field in data discovery protocols, care should be taken to use values which follow conventions for the domain appropriate for the data. For an astronomical object, this may be a name suitable for use within a domain-specific resolution service. Simulated data might also use this sort of name (if simulating a particular object), or a more generic term such as "G2V star".

4.2.2 BaseTarget.description

type: string **type-detail: Section 5.1**
multiplicity: 0..1

Free form description of target.

4.2.3 BaseTarget.position

type: Position **type-detail: Section 6.4.2**

multiplicity: 0..1

This field provides the spatial location of the target. The value is a STC Position object which supports all required dimensionality and coordinate frame specification needs.

4.3 Target

Extension of BaseTarget, this is a general purpose Target object.

4.3.1 Target.objectClass

type: string **type-detail: Section 5.1**
multiplicity: 0..1

General classification or type of the target. This field supports the discovery of data pertaining to a common class of object, e.g. "Star", "Galaxy", "AGN". At the time of this writing, there is no IVOA recommended vocabulary for this field. The SIMBAD and NED databases use defined vocabularies for astronomical object classifications which may serve as the basis for such.

4.4 AstroTarget

Extension of BaseTarget specialized for astronomical objects. The AstroTarget defines additional astronomical properties of the target.

4.4.1 AstroTarget.name

type: string **type-detail: Section 5.1**
multiplicity: 1

When referring to an astronomical target, one may specify a particular object, or a more general target such as the name of a survey field. When specifying a particular object, it is highly recommended to use a name suitable for input to a name resolver.

4.4.2 AstroTarget.position

type: Position **type-detail: Section 6.4.2**
multiplicity: 0..1

In the context of the astronomical target, this field gives the nominal RA and Dec location for the target. For example, the catalog position of the source.

4.4.3 AstroTarget.objectClass

type: string **type-detail: Section 5.1**
multiplicity: 0..1

General classification or type of the target. This field supports the discovery of data pertaining to a common class of object, e.g. "Star", "Galaxy", "AGN". At the time of this writing, there is no IVOA recommended vocabulary for this field. The SIMBAD and NED databases use defined vocabularies for astronomical object classifications which may serve as the basis for such.

4.4.4 AstroTarget.spectralClass

type: string **type-detail: Section 5.1**
multiplicity: 0..1

Spectral class of the object. As with objectClass, there is no IVOA recommended vocabulary for specifying the spectral class of an object. There is an IVOA Note on the subject entitled "An

encoding system to represent stellar spectral classes in archival databases and catalogs"[7], describing an encoding system which has been adopted by the MAST archive.

4.4.5 AstroTarget.redshift

type: real **type-detail: Section 5.1**
multiplicity: 0..1

This field gives the canonical redshift of the astronomical object. It is normally used to store the cosmological redshift of extragalactic objects, although it may also be used to store the observed redshift of Galactic sources if that information is felt by the data provider to be useful.

4.4.6 AstroTarget.varAmpl

type: real **type-detail: Section 5.1**
multiplicity: 0..1

Canonical variability amplitude attributed to the target.

4.5 ObsConfig

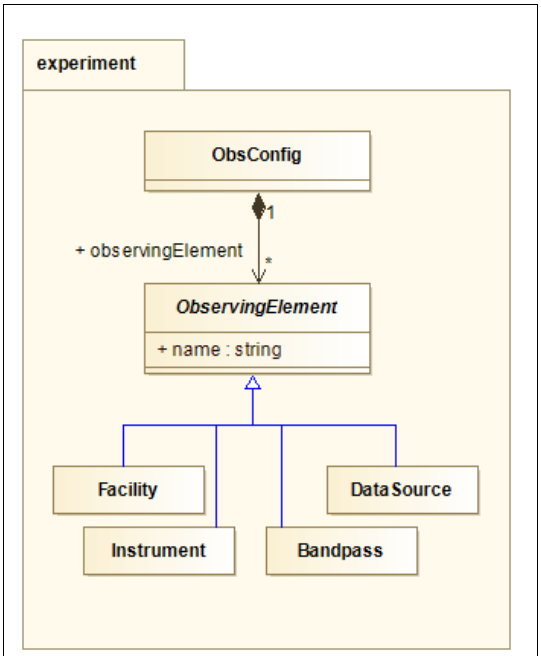


Figure 11: ObsConfig detail diagram

ObsConfig is a simple container object for all Observation Configuration metadata. It is modeled as a simple list of observing elements. Each ObservingElement provides metadata describing a particular domain of the observation setup. For example, the Facility performing the observation, the Instrument being used, etc.

4.5.1 ObsConfig.observingElement

type: ObservingElement **type-detail: Section 4.6**
multiplicity: 0..*

Collection of zero or more ObservingElement objects which define observation parameters.

4.6 ObservingElement

Abstract base class for defining observation parameters. Each subclass of ObservationElement pertains to a particular 'domain' of the observation setup, e.g. the Facility, or the Instrument, and may have additional structure to provide all relevant metadata in that domain.

4.6.1 ObservingElement.name

type: string **type-detail: Section 5.1**
multiplicity: 1

The name attribute identifies the particular instance of the ObservingElement. e.g. the Facility name.

4.6.2 ObservingElement subclasses

There are currently 3 empty extensions of ObservingElement, each identifying a particular domain of Observation Configuration metadata (Facility, Instrument, Bandpass, DataSource). To simplify the text, we merely list the contextual definitions for the name attributes here.

4.6.2.1 Facility.name:

Name of the facility performing the observation.

4.6.2.2 Instrument.name:

This field identifies the instrument used to create the data. (RM:Collection.Instrument) This can be a specific instrument name, general type or something else, such as a program in the case of theoretical data. We restrict this field to a single value.

4.6.2.3 Bandpass.name:

A string describing the spectral range of the observation. The value may be expressed in terms of general spectral bands, or specific bandpass names. If multiple bands are covered, the value may be a comma delimited combination of appropriate bands. If expressed as general bands, the value(s) must be selected from the enumerated set given by the SpectralBand type. There is no controlled vocabulary for bandpass names as the list is too long to enumerate. Effort should be made to use highly recognized bandpass names (eg: "U","V","B","R","I", "H-alpha").

This field corresponds to both the Coverage.Spectral and Coverage.Spectral.Bandpass fields of the Resource Metadata document.

4.6.2.4 DataSource.name:

Describes the original source of the data in a very general fashion. In other words, "What sort of observation originally generated the data?" Suggested values include:

- **survey**: Survey data typically covers some region of observational parameter space with as complete as possible coverage within that region.
- **pointed**: Pointed data of a particular object or field.
- **theory**: Theory data, generated based on a theoretical model.
- **artificial**: Artificial, or simulated data. Similar to 'theory', but not necessarily based on a theoretical model.
- **custom**: Custom data, as part of a specific research project.

4.7 Proposal

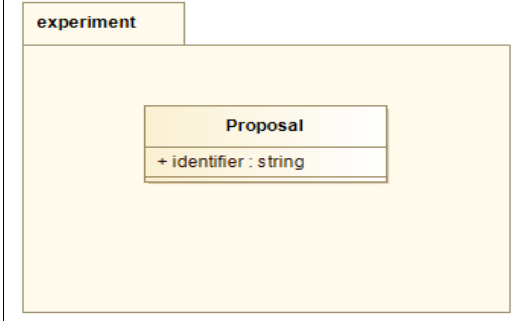


Figure 12: Proposal detail diagram

Metadata related to the proposal or document which spawned the observation.

4.7.1 Proposal.identifier

type: string **type-detail: Section 5.1**
multiplicity: 0..1

Tag used to uniquely identify a particular proposal within the institution or entity.

5 Data Types

5.1 Base Data Types

Provides a set of standardized primitive data types as well as types for representing quantities (values with associated units and ucd). We provide a diagram of the model here, and refer the reader to Appendix E of the VO-DML modeling specification document[8] for more information.

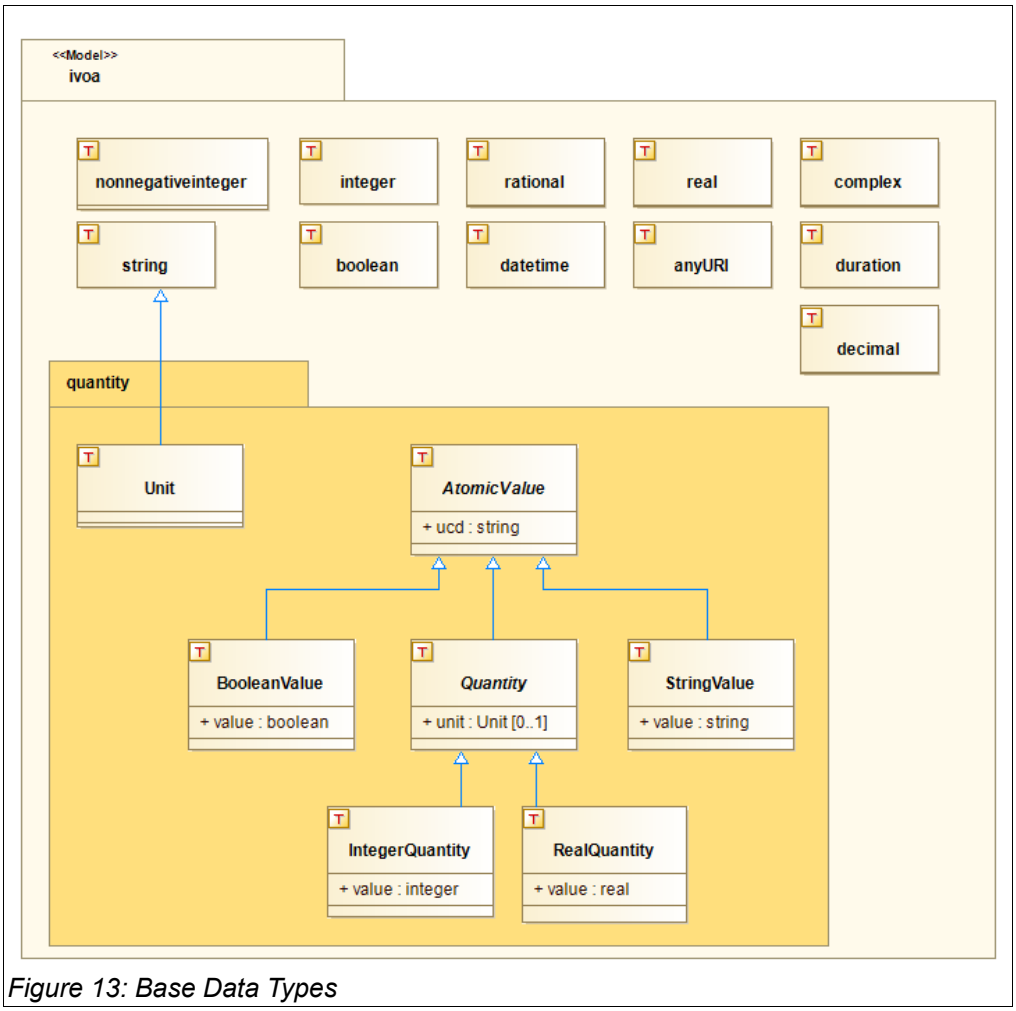


Figure 13: Base Data Types

5.1.1 Units

This model requires the use of the IVOA VOUnits Standard[9] for representing units of physical quantities. This standard reconciles common practices and current standards for use within the IVOA community.

5.1.2 UCDs

This model requires the ucd field to comply with syntax defined in ”An IVOA Standard for Unified Content Descriptors”[11].

5.1.3 Dates

The 'datetime' datatype is for expressing date-time values. The string representation of a datetime 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.

5.2 Dataset Model DataTypes

The Dataset model has gathered and homogenized data type definitions from previous specifications like ObsCore (DataProductType), VOResource (RightsType), and VODataservice (SpectralBandType).

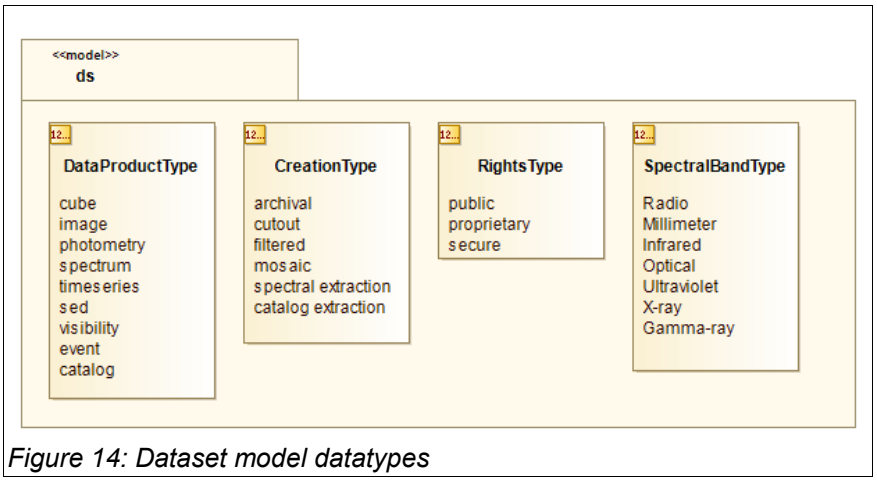


Figure 14: Dataset model datatypes

5.2.1 CreationType

Enumeration of dataset creation types. Allowed values are:

Token	Meaning
archival	Indicates that it is one of a collection of datasets generated in a systematic, homogeneous way and is stored statically (or at least versioned). It will be possible to regenerate this dataset at a later date. The remaining types imply on-the-fly manipulation.
cutout	Indicates that the dataset was created "on-the-fly", by subsetting, but not by modifying values.
filtered	May involve excluding data prior to binning into samples, also "on-the-fly"
mosaic	Combines multiple original datasets "on-the-fly"
spectral extraction	Has been extracted, for example, from a spectral data cube.
catalog extraction	Has been extracted from a catalog.

5.2.2 DataProductType

Enumeration identifying the high level classification of a data product. Allowed values are:

Token	Meaning
cube	A multidimensional astronomical image of three (3) or more axes.
image	A two (2) dimensional astronomical image.
photometry	Dataset with spectral coverage with irregular gaps.
spectrum	Dataset where spectral coverage is the primary attribute, in contiguous bins. e.g. a 1D spectrum or a long slit spectrum.
timeseries	Dataset presenting some quantity varying as a function of time. A light curve is a typical example of a timeseries dataset.
sed	A spectral energy distribution, an advanced data product often produced by combining data from multiple observations.
visibility	A visibility (radio) dataset. Typically this is instrumental data, and is often a complex object containing multiple files or other substructures. A visibility dataset may contain data with spatial, spectral, time, and polarization information for each measured visibility.
event	An event counting dataset (e.g. X-ray). An event dataset may contain data with spatial, spectral, and time information for each measured event.
catalog	A catalog.

5.2.3 RightsType

Enumeration indicating access rights levels. Allowed values are:

Token	Meaning
public	unrestricted, public access is allowed, without authentication.
secure	authenticated, public access is allowed.
proprietary	only proprietary access is allowed with authentication.

5.2.4 SpectralBandType

Enumeration of generic spectral bands:

Token	**Meaning** (λ=wavelength, v=frequency, E=energy)
Radio	λ ≥ 10 mm; v ≤ 30 GHz
Millimeter	0.1 mm ≤ λ ≤ 10 mm; 3000 GHz ≥ v ≥ 30 GHz
Infrared	1 μ ≤ λ ≤ 100 μ
Optical	0.3 μ ≤ λ ≤ 1 μ
Ultraviolet	100 Å ≤ λ ≤ 3000 Å; 1.2 eV ≤ E ≤ 120 eV
X-ray	0.1 Å ≤ λ ≤ 100 Å; 0.12 keV ≤ E ≤ 120 keV
Gamma-ray	E ≥ 120keV

6 STC 2.0 Prototype Data Model

At the time of this writing, the IVOA data model working group is formalizing a revision of the current STC recommendation (1.33). This new STC model is targeted for use within this document and for the N-Dimensional Cube model. We provide here, a prototype definition of the STC-2.0 data model which is a best approximation of the content and structure of this work. The objects and types presented here are sufficient to support the development of this work, but are not comprehensive of the scope and content of the STC model.

This section should be removed upon the completion of the STC-2.0 model, and related documents updated to that specification.

6.1 Physical Coordinate Systems

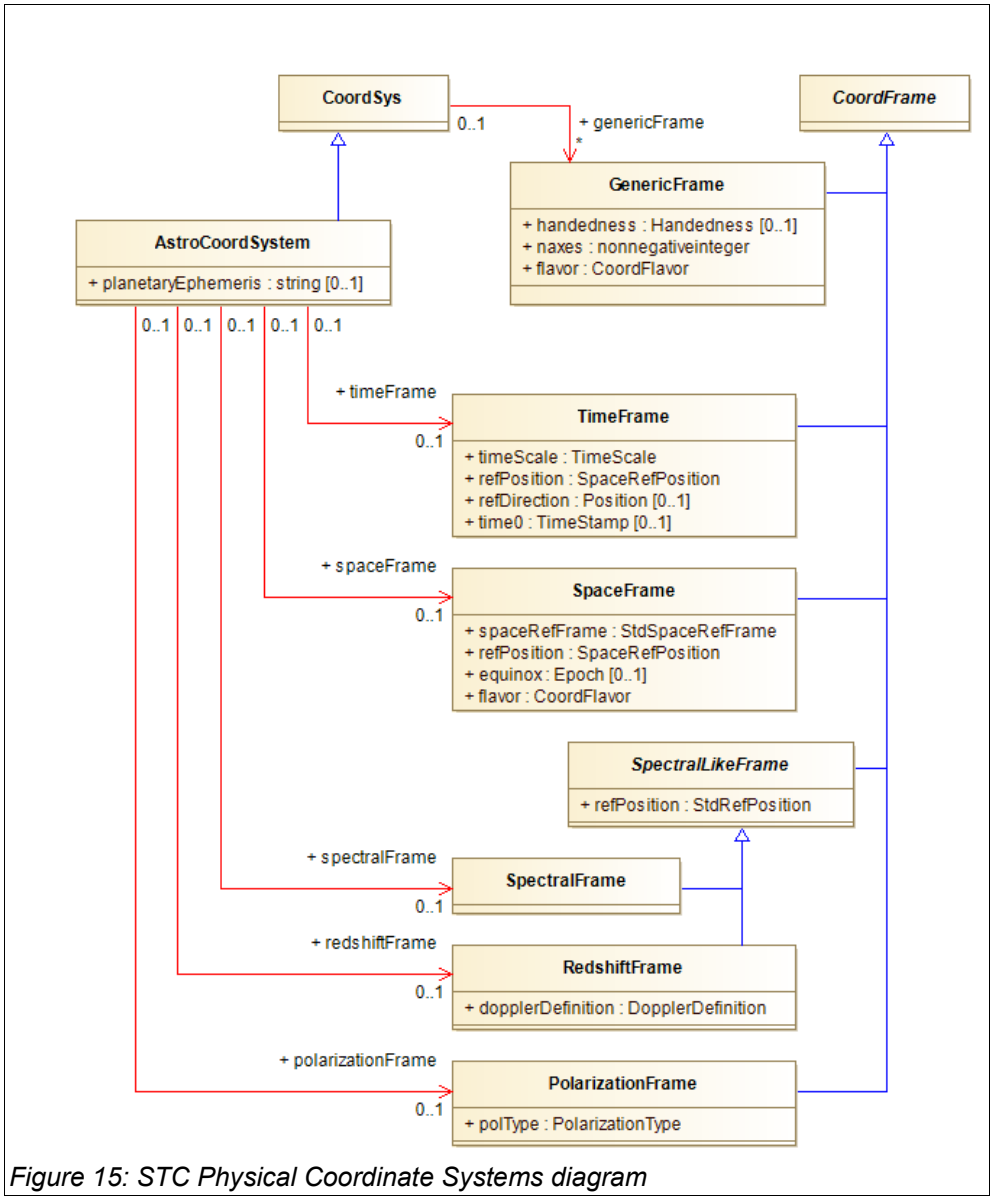


Figure 15: STC Physical Coordinate Systems diagram

In this section we define objects describing physical coordinate systems.

6.1.1 CoordSys

Head of the coordinate system hierarchy for physical systems. Coordinate systems are a collection of coordinate frames which completely define the space in all domains.

6.1.1.1 CoordSys.genericFrame:GenericFrame

type: GenericFrame **type-detail: Section 6.1.4**
multiplicity: 0..*

Zero or more coordinate frames not represented by the specialized frames.

6.1.2 AstroCoordSystem

Extension of CoordSys specialized for astronomical systems. This container holds a reference to zero or one of each specialized domain frame.

6.1.2.1 AstroCoordSystem.planetaryEphemeris

type: string **type-detail: Section 5.1**
multiplicity: 0..1

Required for any position related to a solar system entity other than the geocenter. Default = "DE405"

6.1.2.2 AstroCoordSystem.timeFrame

type: TimeFrame **type-detail: Section 6.1.10**
multiplicity: 0..1

Reference to zero or one Time frame.

6.1.2.3 AstroCoordSystem.spaceFrame

type: SpaceFrame **type-detail: Section 6.1.7**
multiplicity: 0..1

Reference to zero or one Space frame.

6.1.2.4 AstroCoordSystem.spectralFrame

type: SpectralFrame **type-detail: Section 6.1.8**
multiplicity: 0..1

Reference to zero or one Spectral frame.

6.1.2.5 AstroCoordSystem.redshiftFrame

type: RedshiftFrame **type-detail: Section 6.1.6**
multiplicity: 0..1

Reference to zero or one Redshift frame.

6.1.2.6 AstroCoordSystem.polarizationFrame

type: PolarizationFrame **type-detail: Section 6.1.5**
multiplicity: 0..1

Reference to zero or one Polarization frame.

6.1.3 CoordFrame

Abstract head of the Coordinate Frame hierarchy. A Coordinate frame defines the physical domain in which other objects reside. We provide specialized frames for each of the primary

astronomical domains, and a generic frame which may be used for domains not represented here.

6.1.4 GenericFrame

Generic coordinate frame for use describing domains not represented by the specialized frames.

6.1.4.1 GenericFrame.handedness

type: Handedness **type-detail: Section 6.12.4**
multiplicity: 0..1

Handedness of the frame. Values are restricted to the Handedness enumeration list, the default is "left".

6.1.4.2 GenericFrame.naxes

type: nonnegativeinteger **type-detail: Section 5.1**
multiplicity: 1

Number of axes in the domain.

6.1.4.3 GenericFrame.flavor

type: CoordFlavor **type-detail: Section 6.12.1**
multiplicity: 1

Specifies the structure of the frame. Values are restricted to the CoordFlavor enumeration set, (e.g. Cartesian, Spherical, Polar, etc.).

6.1.5 PolarizationFrame

Coordinate frame for the Polarization domain.

6.1.5.1 PolarizationFrame.polType:PolarizationType

type: PolarizationType **type-detail: Section 6.12.5**
multiplicity: 1

Specifies the type of polarization.

6.1.6 RedshiftFrame

This extension of the SpectralLikeFrame serves as the coordinate frame for the Redshift domain.

6.1.6.1 RedshiftFrame.dopplerDefinition

type: DopplerDefinition **type-detail: Section 6.12.2**
multiplicity: 1

Specifies what the definition of redshift is, and how it should be translated to doppler velocity.

6.1.7 SpaceFrame

Coordinate frame for the Spatial domain.

6.1.7.1 SpaceFrame.spaceRefFrame

type: StdSpaceRefFrame **type-detail: Section 6.12.11**
multiplicity: 1

Spatial reference frame specifying the orientation of the frame.

6.1.7.2 SpaceFrame.refPosition

type: SpaceRefPosition **type-detail: Section 6.13.9**
multiplicity: 1

Origin of the spatial coordinate.

6.1.7.3 SpaceFrame.equinox

type: Epoch **type-detail: Section 6.13.1**
multiplicity: 0..1

The epoch for the coordinate system. Fixes the orientation of the system.

6.1.7.4 SpaceFrame.flavor

type: CoordFlavor **type-detail: Section 6.12.1**
multiplicity: 1

Specifies the structure of the frame. Values are restricted to the CoordFlavor enumeration set, (e.g. Cartesian, Spherical, Polar, etc.).

6.1.8 SpectralFrame

This extension of the SpectralLikeFrame serves as the coordinate frame for the Spectral domain. No additional attributes are defined for this frame.

6.1.9 SpectralLikeFrame

Abstract parent for coordinate frames in the overall Spectral domain.

6.1.9.1 SpectralLikeFrame.refPosition

type: StdRefPosition **type-detail: Section 6.12.12**
multiplicity: 1

Origin of the Spectral coordinate frame. For the spectral frame, this must be one of the enumerated standard reference positions.

6.1.10 TimeFrame

Coordinate frame for the Time domain.

6.1.10.1TimeFrame.timeScale

type: TimeScale **type-detail: Section 6.12.13**
multiplicity: 1

IAU recognized time scale.

6.1.10.2TimeFrame.refPosition

type: SpaceRefPosition **type-detail: Section 6.13.9**
multiplicity: 1

Spatial origin of the time coordinate frame.

6.1.10.3TimeFrame.refDirection

type: Position **type-detail: Section 6.4.2**
multiplicity: 0..1

Direction of origin.

6.1.10.4TimeFrame.time0

type: TimeStamp **type-detail: Section 6.13.12**
multiplicity: 0..1

Temporal origin of the time frame. For TimeOffset coordinates, this indicates the time of the triggering event.

6.2 Coordinates

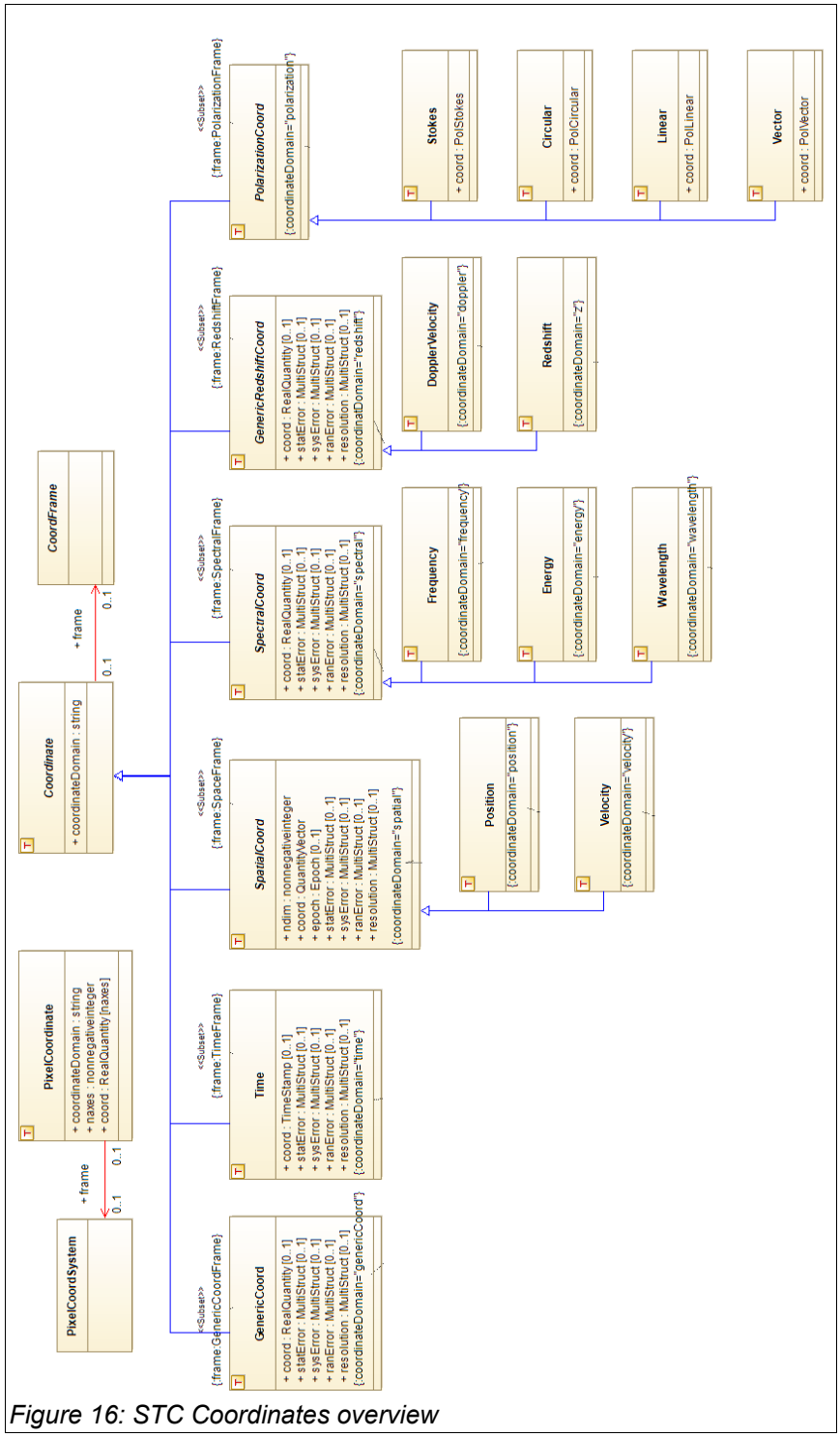


Figure 16: STC Coordinates overview

6.2.1 Coordinate

Abstract head of the Coordinate hierarchy, this object provides a specific location in some domain.

6.2.1.1 Coordinate.coordinateDomain

type: string **type-detail: Section 5.1**
multiplicity: 1

String whose value expresses a skos concept representing the physical domain in which the coordinate belongs. The value is used to define domain specific constraints, (e.g. for units), and ensure consistency between the various attributes (e.g. associating a time frame with a spatial coordinate).

6.2.1.2 Coordinate.frame

type: CoordFrame **type-detail: Section 6.1.3**
multiplicity: 0..1

Reference to zero or one CoordFrame describing the domain of the coordinate.

6.2.2 GenericCoord

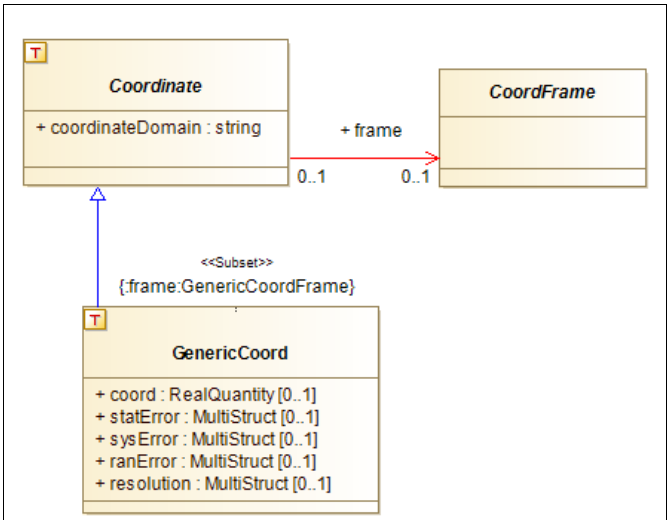


Figure 17: Generic Coordinates diagram

Realizable 1D ScalarCoord for use when a specialized domain type does not apply. The expectation is that only the spatial domain requires more than one dimension.

6.2.2.1 GenericCoord.frame

type: GenericFrame **type-detail: Section 6.1.4**
multiplicity: 0..1

The GenericCoord subsets, or constrains, the frame attribute to the generic frame type. Reference to zero or one GenericFrame describing the domain of the coordinate.

6.2.2.2 GenericCoord.coord

type: RealQuantity **type-detail: Section 5.1**
multiplicity: 0..1

Coordinate value, complete with associated unit and ucd.

6.2.2.3 GenericCoord.statError

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Statistical error associated with the coordinate value, if any, is provided by zero or one MultiStruct objects which facilitate a variety of representations of the error with respect to the value. In this context, the MultiStruct flavor should be set to 'error'.

6.2.2.4 GenericCoord.sysError

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Systematic error associated with the coordinate value, if any, is provided by zero or one MultiStruct objects which facilitate a variety of representations of the error with respect to the value. In this context, the MultiStruct flavor should be set to 'error'.

6.2.2.5 GenericCoord.ranError

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Other random error associated with the coordinate value, if any, is provided by zero or one MultiStruct objects which facilitate a variety of representations of the error with respect to the value. In this context, the MultiStruct flavor should be set to 'error'.

6.2.2.6 GenericCoord.resolution

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Coordinate resolution. In this context, the MultiStruct flavor should be set to 'resolution'. (TBR)

6.3 Time Coordinates

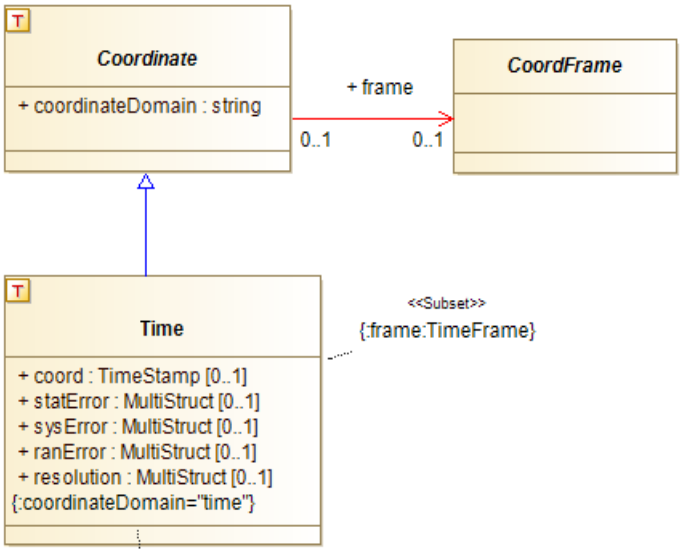


Figure 18: Time Domain Coordinates diagram

6.3.1 Time

Extension of Coordinate type specialized to the Time domain. This class is responsible for ensuring that all quantities and errors are compatible with this domain.

6.3.1.1 Time.frame

type: TimeFrame **type-detail: Section 6.1.10**
multiplicity: 0..1

The TimeCoord subsets, or constrains, the frame attribute to the type appropriate for the Time domain. Reference to zero or one TimeFrame describing the domain of the coordinate.

6.3.1.2 Time.coord

type: TimeStamp **type-detail: Section 6.13.12**
multiplicity: 0..1

Coordinate value indicating an instant in time, using any of the TimeStamp data types.

6.3.1.3 Time.statError

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Statistical error associated with the coordinate value, if any, is provided by zero or one MultiStruct objects which facilitate a variety of representations of the error with respect to the value.

6.3.1.4 Time.sysError

type: MultiStruct **type-detail: Section 6.9.1**

multiplicity: 0..1

Systematic error associated with the coordinate value, if any, is provided by zero or one MultiStruct objects which facilitate a variety of representations of the error with respect to the value.

6.3.1.5 Time.ranError

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Other random error associated with the coordinate value, if any, is provided by zero or one MultiStruct objects which facilitate a variety of representations of the error with respect to the value.

6.3.1.6 Time.resolution

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Coordinate resolution. (TBR)

6.4 Spatial Coordinates

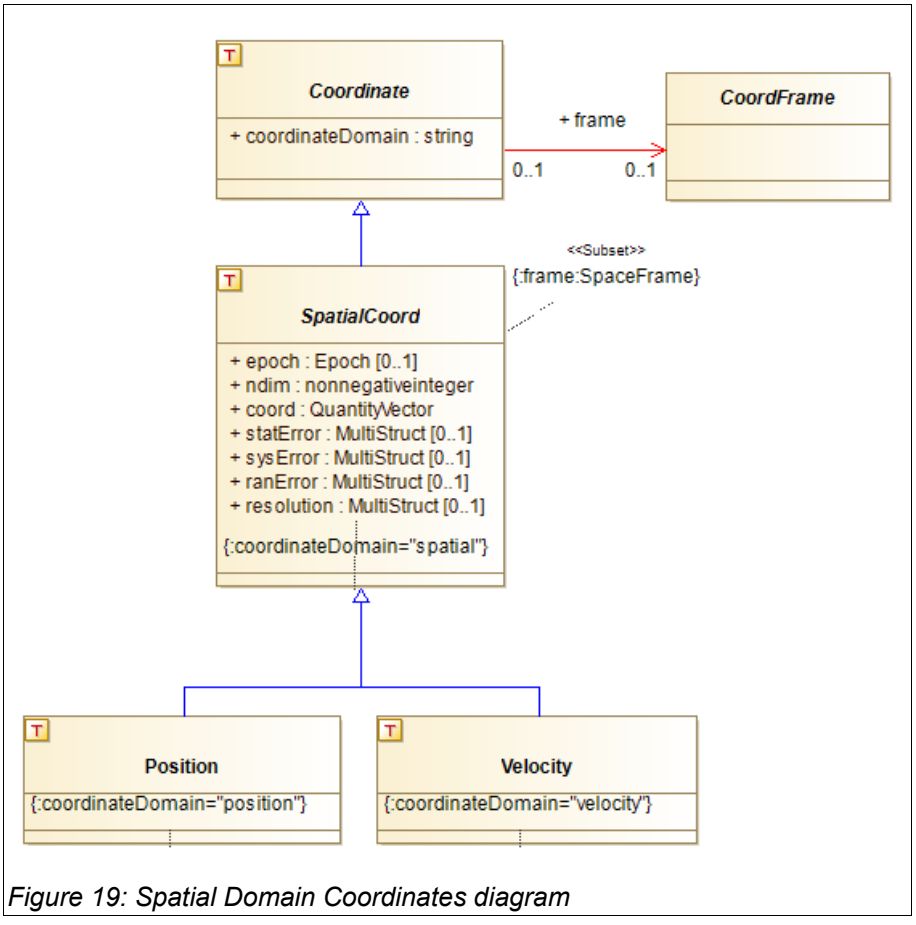


Figure 19: Spatial Domain Coordinates diagram

6.4.1 SpatialCoord

Abstract extension of Coordinate type specialized to the Spatial domain. This class is responsible for ensuring that all quantities and errors are compatible with this domain.

6.4.1.1 SpatialCoord.frame

type: SpaceFrame **type-detail: Section 6.1.7**
multiplicity: 0..1

The SpaceCoord subsets, or constrains, the frame attribute to the type appropriate for the Space domain. Reference to zero or one SpaceFrame describing the domain of the coordinate.

6.4.1.2 SpatialCoord.ndim

type: nonnegativeinteger **type-detail: Section 5.1**
multiplicity: 1

Number of spatial dimensions.

6.4.1.3 SpatialCoord.coord

type: QuantityVector **type-detail: Section 6.13.5**
multiplicity: 1

Coordinate value, complete with associated unit and ucd.

6.4.1.4 SpatialCoord.statError

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Statistical error associated with the coordinate value, if any, is provided by zero or one MultiStruct objects which facilitate a variety of representations of the error with respect to the value. In this context, the MultiStruct flavor should be set to 'error'.

6.4.1.5 SpatialCoord.sysError

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Systematic error associated with the coordinate value, if any, is provided by zero or one MultiStruct objects which facilitate a variety of representations of the error with respect to the value. In this context, the MultiStruct flavor should be set to 'error'.

6.4.1.6 SpatialCoord.ranError

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Other random error associated with the coordinate value, if any, is provided by zero or one MultiStruct objects which facilitate a variety of representations of the error with respect to the value. In this context, the MultiStruct flavor should be set to 'error'.

6.4.1.7 SpatialCoord.resolution

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Coordinate resolution. In this context, the MultiStruct flavor should be set to 'resolution'. (TBR)

6.4.1.8 SpatialCoord.epoch

type: Epoch **type-detail: Section 6.13.1**
multiplicity: 0..1

Epoch associated with the particular coordinate. Note: this is independent of the equinox of the associated coordinate system, which fixes the orientation of the system.

6.4.2 Position

Extension of SpatialCoord for positional coordinates.

6.4.2.1 Position.coordinateDomain

type: string **type-detail: Section 5.1**
multiplicity: 1

The Position subsets, or constrains, the coordinate domain value to the 'position' concept.

6.4.3 Velocity

Extension of SpatialCoord for velocity coordinates (time derivative of position).

6.4.3.1 Velocity.coordinateDomain

type: string **type-detail: Section 5.1**
multiplicity: 1

The Velocity subsets, or constrains, the coordinate domain value to the 'velocity' concept.

6.5 Spectral Coordinates

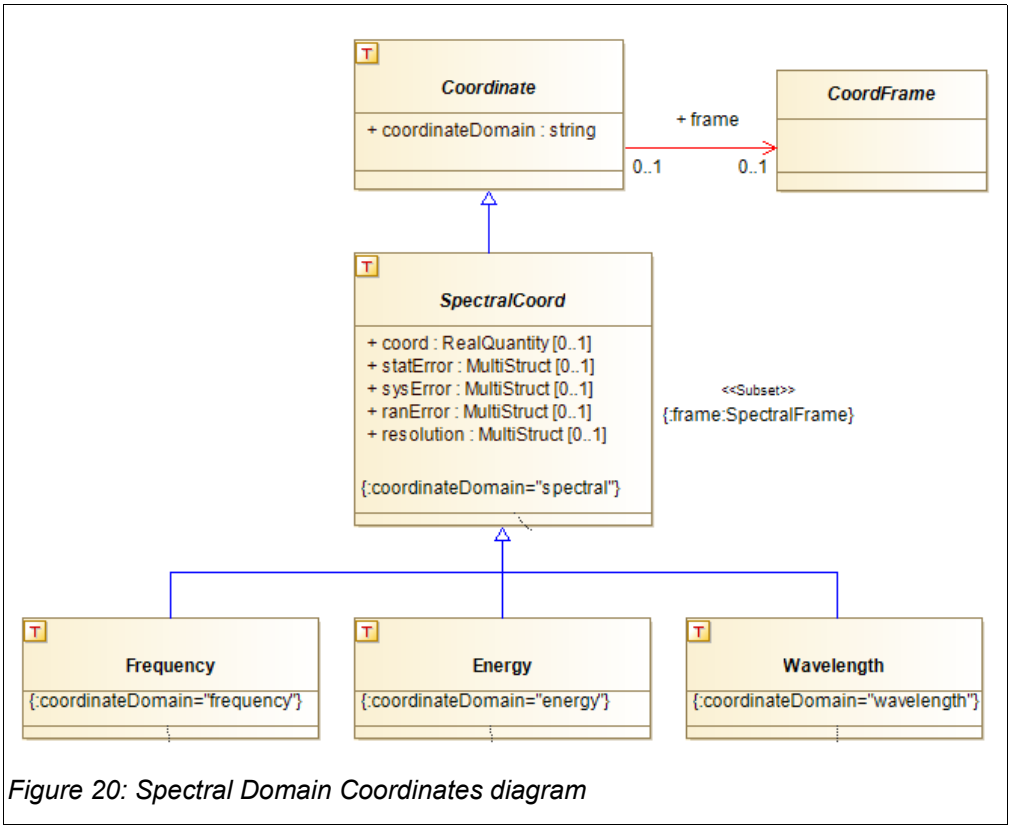


Figure 20: Spectral Domain Coordinates diagram

6.5.1 SpectralCoord

Extension of ScalarCoordinate type specialized to the Spectral domain. This class is responsible for ensuring that all quantities and errors are compatible with this domain.

6.5.1.1 SpectralCoord.coordinateDomain

type: string **type-detail: Section 5.1**
multiplicity: 1

The SpectralCoord subsets, or constrains, the coordinate domain value to the 'spectral' concept.

6.5.1.2 SpectralCoord.frame

type: SpectralFrame **type-detail: Section 6.1.8**
multiplicity: 0..1

The SpectralCoord subsets, or constrains, the frame attribute to the type appropriate for the Spectral domain. Reference to zero or one SpectralFrame describing the domain of the coordinate.

6.5.1.3 SpectralCoord.coord

type: RealQuantity **type-detail: Section 5.1**
multiplicity: 1

Coordinate value, complete with associated unit and ucd.

6.5.1.4 SpectralCoord.statError

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Statistical error associated with the coordinate value, if any, is provided by zero or one MultiStruct objects which facilitate a variety of representations of the error with respect to the value. In this context, the MultiStruct flavor should be set to 'error'.

6.5.1.5 SpectralCoord.sysError

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Systematic error associated with the coordinate value, if any, is provided by zero or one MultiStruct objects which facilitate a variety of representations of the error with respect to the value. In this context, the MultiStruct flavor should be set to 'error'.

6.5.1.6 SpectralCoord.ranError

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Other random error associated with the coordinate value, if any, is provided by zero or one MultiStruct objects which facilitate a variety of representations of the error with respect to the value. In this context, the MultiStruct flavor should be set to 'error'.

6.5.1.7 SpectralCoord.resolution

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Coordinate resolution. In this context, the MultiStruct flavor should be set to 'resolution'. (TBR)

6.5.2 Energy

6.5.2.1 Energy.coordinateDomain

type: string **type-detail: Section 5.1**
multiplicity: 1

The SpectralCoord subsets, or constrains, the coordinate domain value to the 'energy' concept.

6.5.3 Frequency

6.5.3.1 Frequency.coordinateDomain

type: string **type-detail: Section 5.1**
multiplicity: 1

The SpectralCoord subsets, or constrains, the coordinate domain value to the 'frequency' concept.

6.5.4 Wavelength

6.5.4.1 Wavelength.coordinateDomain

type: string **type-detail: Section 5.1**
multiplicity: 1

The SpectralCoord subsets, or constrains, the coordinate domain value to the 'wavelength' concept.

6.6 Redshift Coordinates

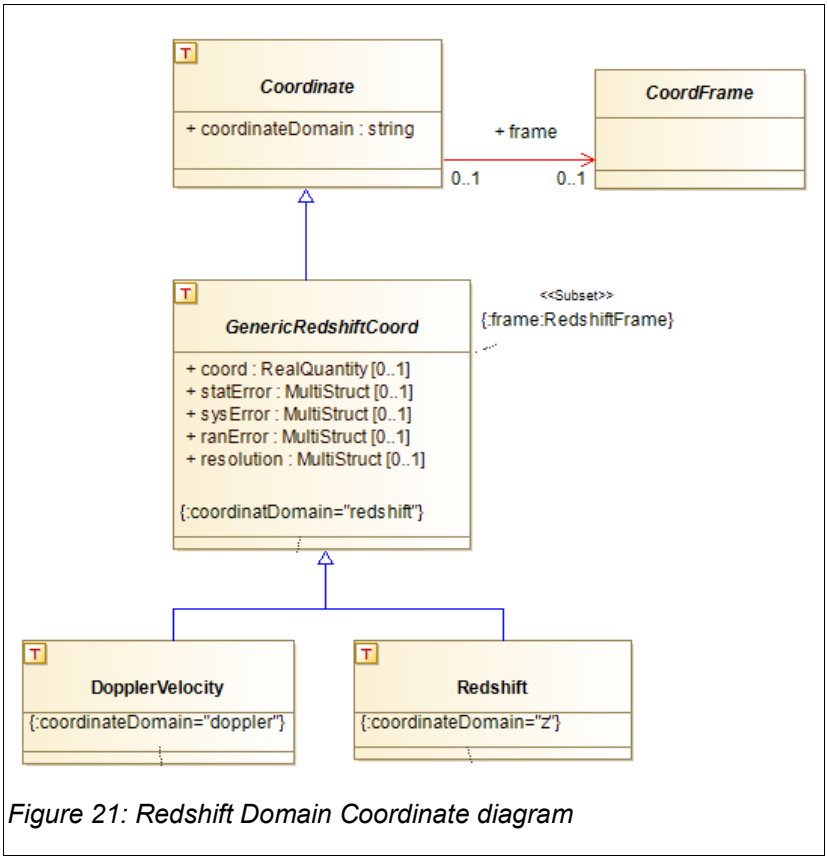


Figure 21: Redshift Domain Coordinate diagram

6.6.1 GenericRedshiftCoord

Extension of ScalarCoordinate type specialized to the Redshift domain. This class is responsible for ensuring that all quantities and errors are compatible with this domain.

6.6.1.1 GenericRedshiftCoord.coordinateDomain

type: string **type-detail: Section 5.1**
multiplicity: 1

The GenericRedshiftCoord subsets, or constrains, the coordinate domain value to the 'redshift' concept.

6.6.1.2 GenericRedshiftCoord.frame

type: RedshiftFrame **type-detail: Section 6.1.6**
multiplicity: 0..1

The GenericRedshiftCoord subsets, or constrains, the frame attribute to the type appropriate for the Redshift domain. Reference to zero or one RedshiftFrame describing the domain of the coordinate.

6.6.1.3 GenericRedshift.coord

type: RealQuantity **type-detail: Section 5.1**
multiplicity: 1

Coordinate value, complete with associated unit and ucd.

6.6.1.4 GenericRedshift.statError

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Statistical error associated with the coordinate value, if any, is provided by zero or one MultiStruct objects which facilitate a variety of representations of the error with respect to the value. In this context, the MultiStruct flavor should be set to 'error'.

6.6.1.5 GenericRedshift.sysError

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Systematic error associated with the coordinate value, if any, is provided by zero or one MultiStruct objects which facilitate a variety of representations of the error with respect to the value. In this context, the MultiStruct flavor should be set to 'error'.

6.6.1.6 GenericRedshift.ranError

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Other random error associated with the coordinate value, if any, is provided by zero or one MultiStruct objects which facilitate a variety of representations of the error with respect to the value. In this context, the MultiStruct flavor should be set to 'error'.

6.6.1.7 GenericRedshift.resolution

type: MultiStruct **type-detail: Section 6.9.1**
multiplicity: 0..1

Coordinate resolution. In this context, the MultiStruct flavor should be set to 'resolution'. (TBR

6.6.2 DopplerVelocity

6.6.2.1 DopplerVelocity.coordinateDomain

type: string **type-detail: Section 5.1**
multiplicity: 1

The DopplerVelocity subsets, or constrains, the coordinate domain value to the 'redshift' concept.

6.6.3 Redshift

6.6.3.1 RedshiftCoord.coordinateDomain

type: string **type-detail: Section 5.1**
multiplicity: 1

The Redshift subsets, or constrains, the coordinate domain value to the 'z' concept.

6.7 Polarization Coordinates

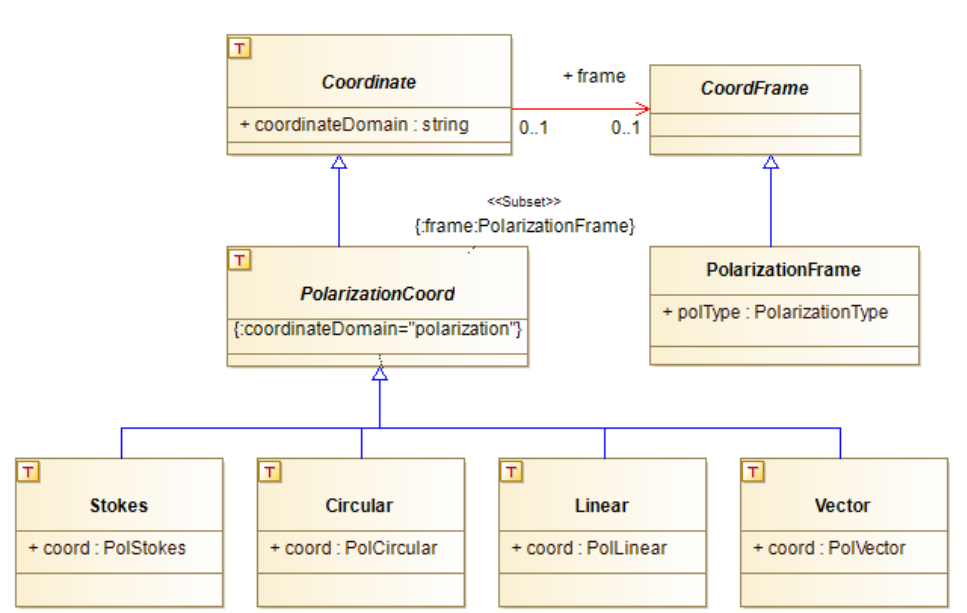


Figure 22: Polarization Domain Coordinates diagram

6.7.1 PolarizationCoord

Abstract head of Coordinate specialized to the Polarization domain. This class is responsible for ensuring that all quantities and errors are compatible with this domain.

6.7.1.1 PolarizationCoord.coordinateDomain

type: string **type-detail: Section 5.1**
multiplicity: 1

The PolarizationCoord subsets, or constrains, the coordinate domain value to the 'polarization' concept.

6.7.1.2 PolarizationCoord.frame

type: PolarizationFrame **type-detail: Section 6.1.5**
multiplicity: 0..1

Reference to zero or one PolarizatonFrame describing the domain of the coordinate.

6.7.2 Circular

Extension of PolarizationCoord specialized for the Circular Polarization type.

6.7.2.1 Circular.coord:PolCircular

type: PolCircular **type-detail: Section 6.12.7**
multiplicity: 0..1

Coordinate value is one of the enumerated set defined by PolCircular

6.7.3 Linear

Extension of PolarizationCoord specialized for the Linear Polarization type.

6.7.3.1 Linear.coord:PolLinear

type: PolLinear **type-detail: Section 6.12.8**
multiplicity: 0..1

Coordinate value is one of the enumerated set defined by PolLinear

6.7.4 Stokes

Extension of PolarizationCoord specialized for the Stokes Polarization type.

6.7.4.1 Stokes.coord:PolStokes

type: PolStokes **type-detail: Section 6.12.6**
multiplicity: 0..1

Coordinate value is one of the enumerated set defined by PolStokes

6.7.5 Vector

Extension of PolarizationCoord specialized for the Vector Polarization type.

6.7.5.1 Vector.coord:PolVector

type: PolVector **type-detail: Section 6.12.9**
multiplicity: 0..1

Coordinate value is one of the enumerated set defined by PolVector

6.8 Pixel Coordinate Systems

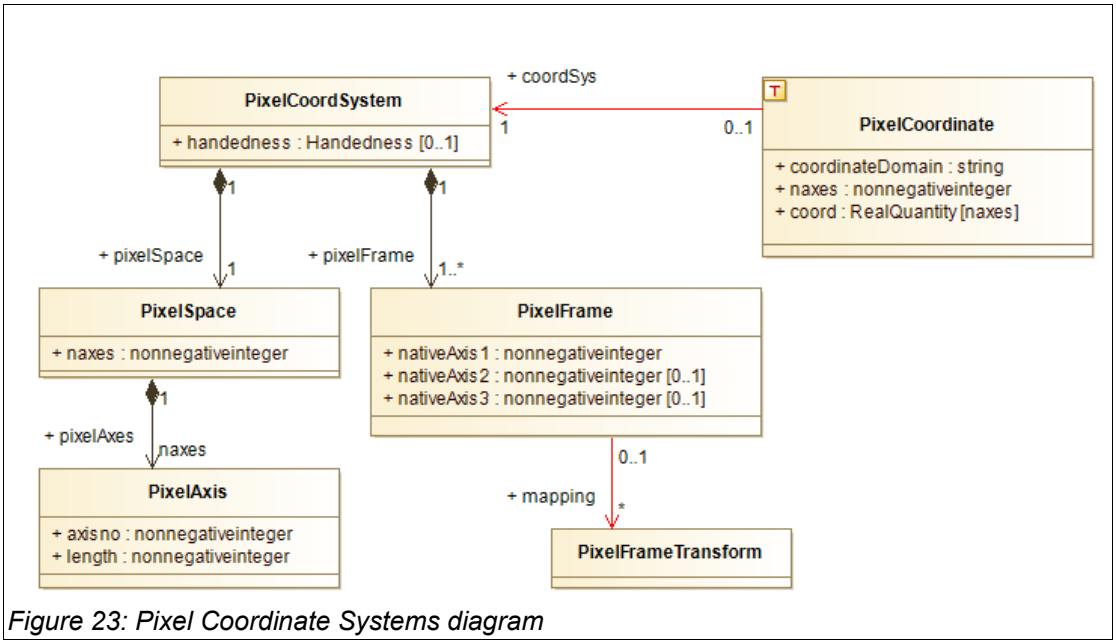


Figure 23: Pixel Coordinate Systems diagram

This section describes objects associated with pixelated coordinate space.

6.8.1 PixelCoordSystem

The PixelCoordSystem holds the complete description of the pixelated coordinate space. The PixelCoordSystem holds a collection of pixel frames, grouping associated axes, and objects describing the number and size of each pixel axis.

6.8.1.1 PixelCoordSystem.handedness

type: Handedness **type-detail: Section 6.12.4**
multiplicity: 0..1

Handedness ("left" or "right") of the coordinate system.

6.8.1.2 PixelCoordSystem.pixelFrame

type: PixelFrame **type-detail: Section 6.8.4**
multiplicity: 1..*

Collection of one or more pixel frames defining the pixel coordinate system. Each PixelFrame represents a set of associated pixel axes.

6.8.1.3 PixelCoordSystem.pixelSpace

type: PixelSpace **type-detail: Section 6.8.2**
multiplicity: 1

Describes the number and span of each pixel axis.

6.8.2 PixelSpace

This object defines the number and size of each pixel axis.

6.8.2.1 PixelSpace.naxes

type: nonnegativeinteger **type-detail: Section 5.1**
multiplicity: 1

Number of pixel axes.

6.8.2.2 PixelSpace.pixelAxes

type: PixelAxis **type-detail: Section 6.8.3**
multiplicity: naxes

Axis number and size. One per axis.

6.8.3 PixelAxis

This object identifies a pixel axis and defines its size.

6.8.3.1 PixelAxis.naxis

type: nonnegativeinteger **type-detail: Section 5.1**
multiplicity: 1

Axis number

6.8.3.2 PixelAxis.numpix

type: nonnegativeinteger **type-detail: Section 5.1**
multiplicity: 1

Length of the pixel axis.

6.8.4 PixelFrame

Specifies a set of pixel axes which are considered a unit. For example, 2 pixel axes which map to a 2-D SpaceFrame. A PixelFrame can have, at most, 3 dimensions.

6.8.4.1 PixelFrame.mapping

type: PixelFrameTransform **type-detail: Section 6.10.3**
multiplicity: 0..*

References to mappings, each mapping defines the transform from this PixelFrame to some physical coordinate frame.

6.8.4.2 PixelFrame.nativeAxis1

type: nonnegativeinteger **type-detail: Section 5.1**
multiplicity: 1

This specifies which pixel axis of the pixel space corresponds to dimension 1 of the pixel frame.

6.8.4.3 PixelFrame.nativeAxis2

type: nonnegativeinteger **type-detail: Section 5.1**
multiplicity: 0..1

This specifies which pixel axis of the pixel space corresponds to dimension 1 of the pixel frame.

6.8.4.4 PixelFrame.nativeAxis3

type: nonnegativeinteger **type-detail: Section 5.1**
multiplicity: 0..1

This specifies which pixel axis of the pixel space corresponds to dimension 1 of the pixel frame.

6.8.5 PixelCoordinate

PixelCoordinate provides a complete nD location in pixel space. While a physical coordinate has a reference to an associated physical coordinate frame, a pixel coordinate refers back to the full pixel coordinate system.

6.8.5.1 PixelCoordinate.coordinateDomain

type: string **type-detail: Section 5.1**
multiplicity: 1

String indicating the coordinate domain, for pixel coordinates, the value must be 'pixel'.

6.8.5.2 PixelCoordinate.coordSys

type: PixelCoordSystem **type-detail: Section 6.8.1**
multiplicity: 0..1

Reference to the complete pixel coordinate system. This facilitates access to the frame information and transforms to various physical coordinate systems.

6.8.5.3 PixelCoordinate.naxes

type: nonnegativeinteger **type-detail: Section 5.1**
multiplicity: 1

Dimensionality of the pixel coordinate. Must equal the dimensionality of the associated pixel space.

6.8.5.4 PixelCoordinate.coord

type: RealQuantity **type-detail: Section 5.1**
multiplicity: naxes

Coordinate value in each dimension, ordered according to the pixel space axis definitions.

6.9 MultiStruct

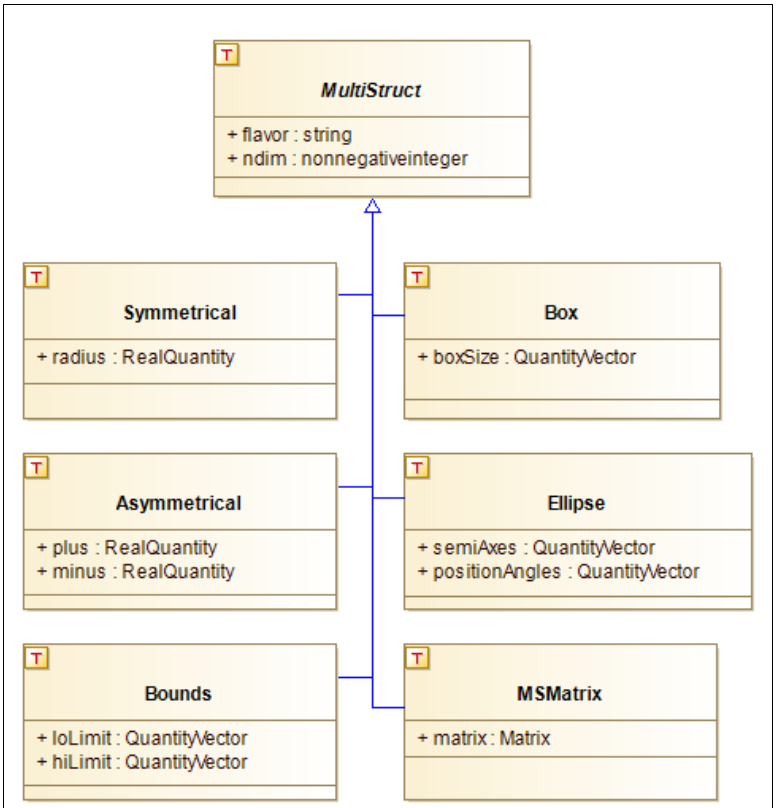


Figure 24: STC MultiStruct

6.9.1 MultiStruct

Abstract head of the MultiStruct hierarchy. This object set represents a multiple-purpose structure which is used to represent Errors, Resolutions, and Sizes in this model.

6.9.1.1 MultiStruct.flavor

type: string **type-detail: Section 5.1**
multiplicity: 1

String representing the role that the object plays in the associated object. e.g. "error", "resolution", etc.

6.9.2 Asymmetrical

The entity is represented by an asymmetrical distribution about the associated value.

6.9.2.1 Asymmetrical.plus

type: RealQuantity **type-detail: Section 5.1**
multiplicity: 1

Relative extension in the positive direction from the associated value. default = Inf.

6.9.2.2 Asymmetrical.minus

type: RealQuantity **type-detail: Section 5.1**
multiplicity: 1

Relative extension in the negative direction from the associated value. default = Inf.

6.9.3 Bounds

The entity is represented by a range.

6.9.3.1 Bounds.loLimit

type: QuantityVector **type-detail: Section 6.13.5**
multiplicity: 1

Minimum value of the range.

6.9.3.2 Bounds.hiLimit

type: QuantityVector **type-detail: Section 6.13.5**
multiplicity: 1

Maximum value of the range.

6.9.4 Symmetrical

The entity is represented by a symmetrical distribution about the associated value.

6.9.4.1 Symmetrical.radius

type: RealQuantity **type-detail: Section 5.1**
multiplicity: 1

Relative extension from the associated value in all directions.

6.9.5 Box

The entity is represented by a n-D box.

6.9.5.1 Box.boxSize

type: QuantityVector **type-detail: Section 6.13.5**
multiplicity: 1

Size of box in each dimension.

6.9.6 Ellipse

The entity is represented by an ellipse or ellipsoid.

6.9.6.1 Ellipse.semiAxes

type: QuantityVector **type-detail: Section 6.13.5**

multiplicity: 1

Size of each semi-axis.

6.9.6.2 Ellipse.positionAngles

type: QuantityVector **type-detail: Section 6.13.5**
multiplicity: 1

Position angles of the proscribed ellipse.

6.9.7 MSMatrix

The entity is represented by a n-D Matrix.

6.9.7.1 MSMatrix.matrix

type: Matrix **type-detail: Section 6.13.2**
multiplicity: 1

matrix elements.

6.10 Transforms

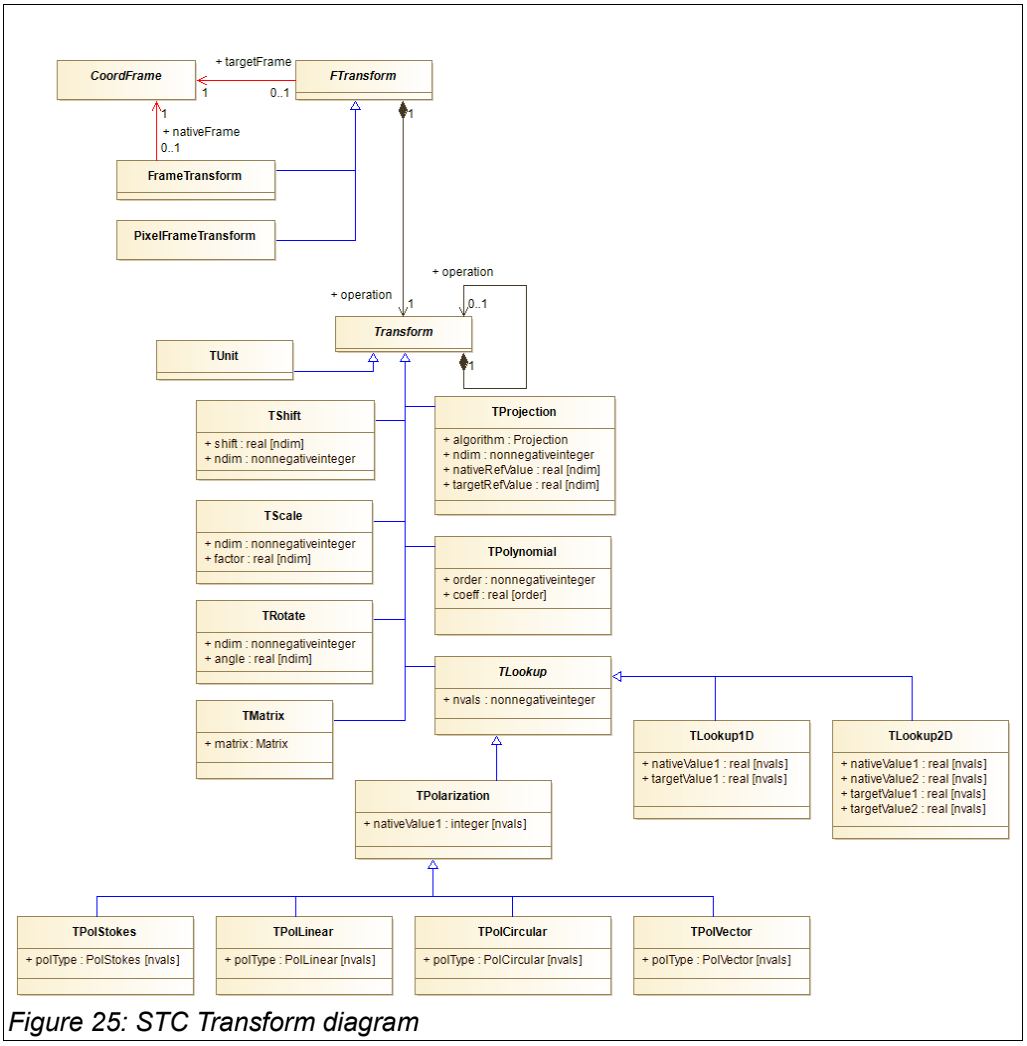


Figure 25: STC Transform diagram

Transforms define the relationship between two coordinate frames. They are used in the conversion of values (Coordinates, Regions, etc) represented in one frame, to the other.

6.10.1 FTransform

Abstract head of the set of bridge classes which define the relationship between two coordinate frames. It identifies the coordinate frames being related, and the operation specification (Transforms) describing the relationship.

6.10.1.1 FTransform.targetFrame

type: CoordFrame **type-detail: Section 6.1.3**
multiplicity: 1

Reference to the coordinate frame which is the target or destination frame. Any given coordinate frame may be the target of multiple relations.

6.10.1.2 FTransform.operation

type: Transform **type-detail: Section 6.10.4**
multiplicity: 1

Operation specification(s) defining the relation between two coordinate frames.

6.10.2 FrameTransform

Extension of FTransform for use in expressing the relation between two physical coordinate frames (CoordFrame).

6.10.2.1 FrameTransform.nativeFrame

type: CoordFrame **type-detail: Section 6.1.3**
multiplicity: 1

Reference to the coodinate frame which is the source, or originating frame.

6.10.3 PixelFrameTransform

Extension of FTransform for use in expressing the relation between a pixel frame and a physical coordinate frame. This object is closely linked with a particular PixelFrame, so there is no need to identify the originating frame.

6.10.4 Transform

Transforms define, provide specifications for, mathematical operations on input objects (e.g. Coordinates), but have no direct knowledge of the physical context of the object. The transform operations defined here are generalized components which can be combined to support a wide range of complex transforms. This includes, but is not limited to, the FITS Linear and WCS coordinate transforms.

6.10.4.1 Transform.operation

type: Transform **type-detail: Section 6.10.4**
multiplicity: 0..1

Enables chaining of Transforms into an ordered sequence of operations in order to build complex transforms from the simpler components.

6.10.5 TUnit

Unit transform. Multiplication by 1.

6.10.6 TShift

Shift operation.

6.10.6.1 TShift.ndim

type: nonnegativeinteger **type-detail: Section 5.1**
multiplicity: 1

Number of dimensions.

6.10.6.2 TShift.shift

type: real **type-detail: Section 5.1**
multiplicity: ndim

Amount to shift in each dimension.

6.10.7 TScale

Multiplication operation

6.10.7.1 TScale.ndim

type: nonnegativeinteger **type-detail: Section 5.1**
multiplicity: 1

Number of dimensions.

6.10.7.2 TScale.factor

type: real **type-detail: Section 5.1**
multiplicity: ndim

Scale factor in each dimension.

6.10.8 TRotate

Angular rotation operation. NOTE: this definition is over-simplified and needs to be fleshed out with specifics about whether these are rotation angles about each axis, or a constant rotation for axis pair, etc.

6.10.8.1 TRotate.ndim

type: nonnegativeinteger **type-detail: Section 5.1**
multiplicity: 1

Number of dimensions.

6.10.8.2 TRotate.angle

type: real **type-detail: Section 5.1**
multiplicity: ndim

Angle of rotation.

6.10.9 TMatrix

Matrix operation.

6.10.9.1 TMatrix.matrix

type: Matrix **type-detail: Section 6.13.2**
multiplicity: 1

6.10.10 TProjection

WCS Projection operation.

6.10.10.1 TProjection.algorithm

type: Projection **type-detail: Section 6.12.10**

multiplicity: 1

Projection algorithm to apply.

6.10.10.2 TProjection.ndim

type: nonnegativeinteger **type-detail: Section 5.1**
multiplicity: 1

Dimensionality of values.

6.10.10.3 TProjection.nativeRefValue

type: real **type-detail: Section 5.1**
multiplicity: ndim

Reference value for originating frame (e.g. Reference Pixel)

6.10.10.4 TProjection.targetRefValue

type: real **type-detail: Section 5.1**
multiplicity: ndim

Reference value at destination frame (e.g. Reference Value)

6.10.11 TPolynomial

1-D Polynomial operation

6.10.11.1 TPolynomial.order

type: nonnegativeinteger **type-detail: Section 5.1**
multiplicity: 1

Polynomial order or degree

6.10.11.2 TPolynomial.coeff

type: real **type-detail: Section 5.1**
multiplicity: order+1

Polynomial coefficients.

6.10.12 TLookup

Abstract head of Lookup table operations.

6.10.12.1 TLookup.nvals

type: nonnegativeinteger **type-detail: Section 5.1**
multiplicity: 1

Number of lookup entries provided (length of table).

6.10.13 TLookup1D

Lookup table in one dimension. Results are determined by linear interpolation of nearest neighbors in the provided lookup table data.

6.10.13.1 TLookup1D.nativeValue1

type: real **type-detail: Section 5.1**
multiplicity: nvals

Originating frame values.

6.10.13.2 TLookup1D.targetValue1

type: real **type-detail: Section 5.1**
multiplicity: nvals

Destination frame values.

6.10.14 TLookup2D

Lookup table in two dimensions. Results are determined by linear interpolation of nearest neighbors in the provided lookup table data.

6.10.14.1 TLookup2D.nativeValue1

type: real **type-detail: Section 5.1**
multiplicity: nvals

Originating frame values for axis 1.

6.10.14.2 TLookup2D.nativeValue2

type: real **type-detail: Section 5.1**
multiplicity: nvals

Originating frame values for axis 2.

6.10.14.3 TLookup2D.targetValue1

type: real **type-detail: Section 5.1**
multiplicity: nvals

Destination frame values for axis 1.

6.10.14.4 TLookup2D.targetValue2

type: real **type-detail: Section 5.1**
multiplicity: nvals

Destination frame values for axis 2.

6.10.15 TPolarization

Extension of TLookup, specialized for polarization data.

6.10.15.1 TPolarization.nativeValue1

type: integer **type-detail: Section 5.1**
multiplicity: nvals

Numerical value to associate with a polarization state.

Two use cases are considered:

1) A numerical value is assigned for each polarization state. For example, the FITS-3.0 "Conventional Stokes values" which assign the numbers (1..4,-1..-8) to various polarization

states. In this case, nvals is the number of such states and the nativeValue1 array holds the associated value.

2) A simple list of pixels, where a polarization state is assigned to each pixel. For example, an image with polarization axis, where each index corresponds to a different polarization state. In this case, nvals is the number of pixels, and the nativeValue1 array holds the pixel indexes.

6.10.16 TPolStokes

Polarization lookup where results are Stokes polarization states.

6.10.16.1 TPolStokes.polType

type: PolStokes **type-detail: Section 6.12.6**
multiplicity: nvals

6.10.17 TPolLinear

Polarization lookup where results are Linear polarization states.

6.10.17.1 TPolLinear.polType

type: PolLinear **type-detail: Section 6.12.8**
multiplicity: nvals

6.10.18 TPolCircular

Polarization lookup where results are Circular polarization states.

6.10.18.1 TPolCircular.polType

type: PolCircular **type-detail: Section 6.12.7**
multiplicity: nvals

6.10.19 TPolVector

Polarization lookup where results are Vector polarization states.

6.10.19.1 TPolVector.polType

type: PolVector **type-detail: Section 6.12.9**
multiplicity: nvals

6.11 Mappings

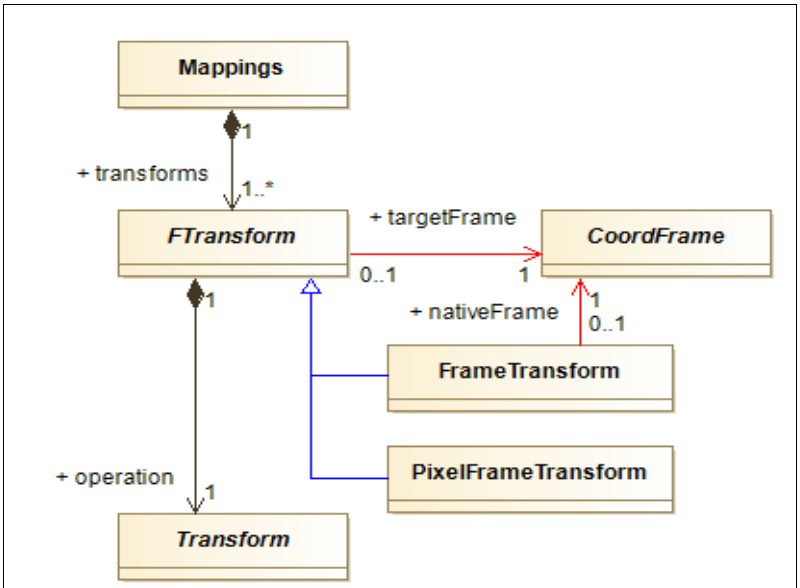


Figure 26: Mappings detail

The Mappings object is a container composed of a set of Frame transforms. This object provides a container to encapsulate and manage a set of coordinate frame transforms associated with a particular object, such as a 2D image. It facilitates the transfer and sharing of this information between applications.

6.11.1 Mappings.transforms

The Mappings container stores one or more FTransform objects, each of which defines a relation between two coordinate frames.

6.12 STC Model Enumerations

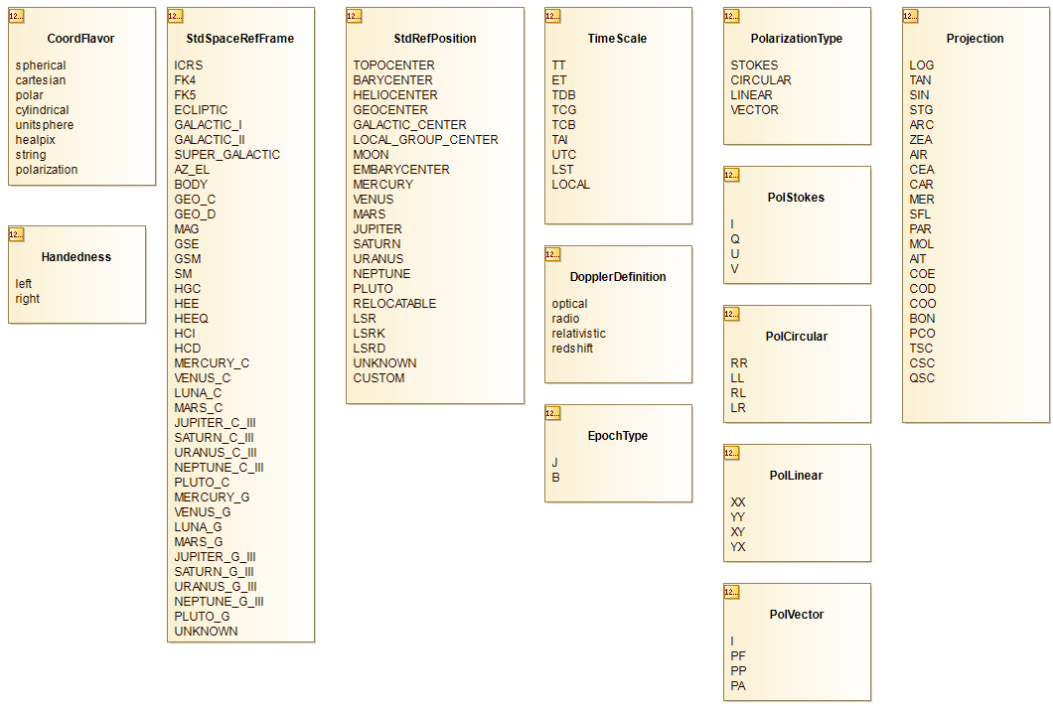


Figure 27: STC Enumerations

6.12.1 CoordFlavor

Token	Meaning
spherical	Spherical 2-D (long,lat) or 3-D (long, lat, rad/elev)
cartesian	Cartesian 1-, 2-, or 3-D coordinates
polar	2-D polar coordinates (radius, posangle)
cylindrical	3-D cylindrical coordinates (radius, posangle, z)
unitsphere	3-D Unit sphere coordinates (direction cosines)
healpix	2-D Healpix coordinates
string	String coordinates
polarization	Polarization coordinates

6.12.2 DopplerDefinition

Token	Meaning
optical	
radio	

relativistic	
redshift	

6.12.3 EpochType

Token	Meaning
J	Julian
B	Besselian

6.12.4 Handedness

Token	Meaning
left	Left handed
right	Right handed

6.12.5 PolarizationType

Enumeration of polarization types.

Token	Meaning
STOKES	
CIRCULAR	
LINEAR	
VECTOR	

6.12.6 PolStokes

Enumeration of Stokes polarization states.

Token	Meaning
I	Standard Stokes unpolarized
Q	Standard Stokes linear
U	Standard Stokes linear
V	Standard Stokes circular

6.12.7 PolCircular

Enumeration of Circular polarization states.

Token	Meaning
RR	Right-right circular
LL	Left-left circular
RL	Right-left cross-circular
LR	Left-right cross-circular

6.12.8 PolLinear

Enumeration of Linear polarization states.

Token	Meaning
XX	X parallel linear
YY	Y parallel linear
XY	XY cross linear
YX	YX cross linear

6.12.9 PolVector

Enumeration of Vector polarization states.

Token	Meaning
I	
PF	
PP	
PA	

6.12.10 Projection

Enumeration of WCS Projection types.

Token	Meaning
LOG	Linear to logarithmic
TAN	Tangent plane projection
SIN	Sine projection
STG	Stereographic projection
ARC	Zenithal equidistant projection
ZEA	Zenithal equal-area projection
AIR	Airy projection
CEA	Cylindrical equal-area projection
CAR	Plate Carre projection
MER	Mercator projection
SFL	Sanson-Flamsteed projection
PAR	Parabolic projection
MOL	Mollweide projection
AIT	Hammer-Aitoff projection
COE	Conic equal-area projection
COD	Conic equidistant projection
COO	Conic orthomorphic projection
BON	Bonne equal-area projection
PCO	Polyconic projection
TSC	Tangential spherical cube projection

CSC	COBE quadrilatealized spherical cube projection
QSC	Quadrilateralized spherical cube projection

6.12.11 StdSpaceRefFrame

Enumeration of standard Space Reference Frames.

Token	Meaning
ICRS	The ICRS frame
FK4	FK4
FK5	FK5
ECLIPTIC	Ecliptic l,b
GALACTIC_I	Old galactic LI, BI
GALACTIC_II	Galactic LII, BII
SUPER_GALACTIC	SGL, SGB
AZ_EL	Azimuth and elevation
BODY	Generic Body (e.g. planet)
GEO_C	Geocentric corotating
GEO_D	Geodetic ref frame
MAG	Geomagnetic ref frame
GSE	Geocentric Solar Ecliptic
GSM	Geocentric Solar Magnetic
SM	Solar Magnetic
HGC	Heliographic
HEE	Heliocentric Earth Ecliptic
HEEQ	Heliocentric Earth Equatorial
HCI	Heliocentric Inertial
HCD	Heliocentric of Date
MERCURY_C	Corotating planetocentric
VENUS_C	Corotating planetocentric
LUNA_C	Corotating planetocentric
MARS_C	Corotating planetocentric
JUPITER_C_III	Corotating planetocentric
SATURN_C_III	Corotating planetocentric
URANUS_C_III	Corotating planetocentric
NEPTUNE_C_III	Corotating planetocentric
PLUTO_C	Corotating planetocentric
MERCURY_G	Corotating planetographic
VENUS_G	Corotating planetographic
LUNA_G	Corotating planetographic
MARS_G	Corotating planetographic
JUPITER_G_III	Corotating planetographic
SATURN_G_III	Corotating planetographic

URANUS_G_III	Corotating planetographic
NEPTUNE_G_III	Corotating planetographic
PLUTO_G	Corotating planetographic
UNKNOWN	Unknown frame
CUSTOM	Custom frame

6.12.12 StdRefPosition

Enumeration of Standard Reference Positions.

Token	Meaning	Note
TOPOCENTER	Location of the observing device	(telescope)
BARYCENTER	Solar system barycenter	
HELIOCENTER	Center of the Sun	
GEOCENTER	Center of the Earth	
GALACTIC_CENTER	Center of the Galaxy	
LOCAL_GROUP_CENTER	Barycenter of the Local Group	
MOON	Center of the Moon	
EMBARYCENTER	Earth-Moon barycenter	
MERCURY	Center of Mercury	
VENUS	Center of Venus	
MARS	Center of Mars	
JUPITER	Center of Jupiter	
SATURN	Center of Saturn	
URANUS	Center of Uranus	
NEPTUNE	Center of Neptune	
PLUTO	Center of Pluto	
RELOCATABLE	Relative origin	Suitable for simulations
LSR	Local Standard of Rest	Spectral/Redshift domain only
LSRK	Kinematic Local Standard of Rest	Equivalent to LSR
LSRD	Dynamic Local Standard of Rest	Spectral/Redshift domain only
UNKNOWN	Unknown origin	
CUSTOM	Origin specified wrt another system	

6.12.13 TimeScale

Enumeration of time scales.

Token	Meaning
LOCAL	Relocatable (simulation) time
TT	Terrestrial Time
UTC	Coordinated Universal Time
ET	Ephemeris Time
TDB	Barycentric Dynamical Time

TCG	Terrestrial Coordinate Time
TCB	Barycentric Coordinate Time
TAI	International Atomic Time
LST	Local Sidereal Time

6.13 STC Model Base Types

Base data types defined by the STC model.

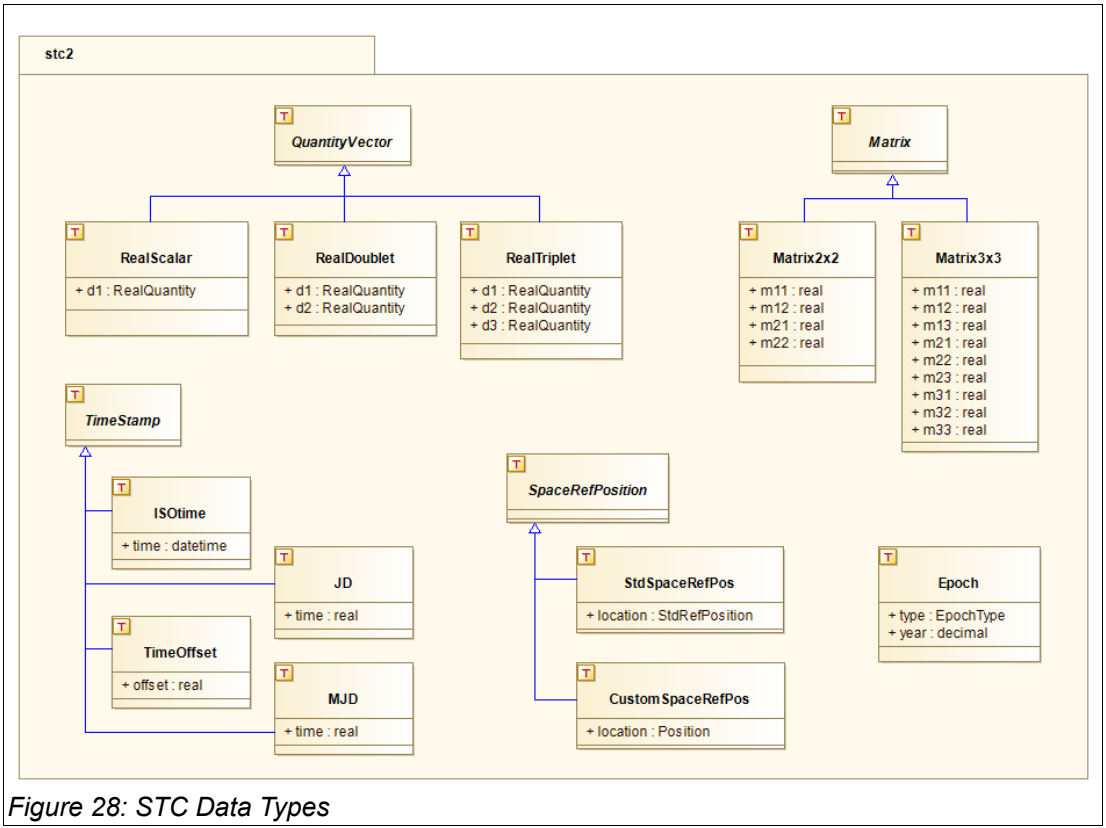


Figure 28: STC Data Types

6.13.1 Epoch

Type for expressing an astronomical epoch date (e.g. "J2000").

6.13.1.1 type

type: EpochType **type-detail: Section 6.12.3**
multiplicity: 1

Basis of the epoch, values must be selected from the EpochType enumeration set.

6.13.1.2 year

type: decimal **type-detail: Section 5.1**
multiplicity: 1

Epoch date expressed as a decimal (e.g. 2000.0)

6.13.2 Matrix

Abstract head of the matrix classes.

6.13.3 Matrix2x2

Basic 2 x 2 matrix. Contains attribute for each matrix element as real type.

6.13.4 Matrix3x3

Basic 3 x 3 matrix. Contains attribute for each matrix element as real type.

6.13.5 QuantityVector

Abstract head of vector types. This type is intended to be used when it is important to emphasize the dimensionality of a value pair, rather than using an array representation. Each component element is independent, and holds their own values for ucd and unit.

6.13.6 RealDoublet

Type for doublet vector. Holds a pair of RealQuantity types.

6.13.6.1 RealDoublet.d1

type: RealQuantity **type-detail: Section 5.1**
multiplicity: 1

First value pair element.

6.13.6.2 RealDoublet.d2

type: RealQuantity **type-detail: Section 5.1**
multiplicity: 1

Second value pair element.

6.13.7 RealScalar

Type for a singluar Quantity vector.

6.13.7.1 RealScalar.d1

type: RealQuantity **type-detail: Section 5.1**
multiplicity: 1

Scalar value element.

6.13.8 RealTriplet

Type for a triplet vector. Hold a three RealQuantity types.

6.13.8.1 RealTriplet.d1

type: RealQuantity **type-detail: Section 5.1**
multiplicity: 1

First triplet element.

6.13.8.2 RealTriplet.d2

type: RealQuantity **type-detail: Section 5.1**
multiplicity: 1

Second triplet element.

6.13.8.3 RealTriplet.d3

type: RealQuantity **type-detail: Section 5.1**
multiplicity: 1

Third triplet element.

6.13.9 SpaceRefPosition

Abstract head of the reference position data type. Spatial reference positions may be described either as an explicit spatial coordinate, or using one of the standard reference position locations. This type facilitates the use of either form in objects.

6.13.10 CustomRefPosition

Custom reference position, represented by a spatial coordinate.

6.13.10.1 CustomRefPosition.location

type: Position **type-detail: Section 6.4.2**
multiplicity: 1

Spatial coordinates for the origin of a Coordinate Frame.

6.13.11 StdSpaceRefPos

Reference position expressed as a known standard origin.

6.13.11.1 StdSpaceRefPos.location

type: StdRefPosition **type-detail: Section 6.12.12**
multiplicity: 1

Origin location selected from the list of standard reference positions.

6.13.12 TimeStamp

Type to express an absolute instant in time.

6.13.13 ISOtime

A time instant expressed as a subset of the ISO-8601 format.
ie: a string of format "yyyy-mm-ddThh:mm:ss.sss"

6.13.13.1 ISOtime.time

type: datetime **type-detail: Section 5.1.3**
multiplicity: 1

ISO time instant value.

6.13.14 JD

Julian day of a particular instant.

6.13.14.1 JD.time

type: real **type-detail: Section 5.1**
multiplicity: 1

Decimal number representing the Julian Day of a time instant, to required precision.

6.13.15 MJD

Modified Julian Day of a particular instant. Note: MJD = JD - 2,400,000.5.

6.13.15.1 MJD.time

type: real **type-detail: Section 5.1**
multiplicity: 1

Decimal number representing the Modified Julian Day of a time instant, to required precision.

6.13.16 TimeOffset

Elapsed time since a particular instant.

6.13.16.1 TimeOffset.offset

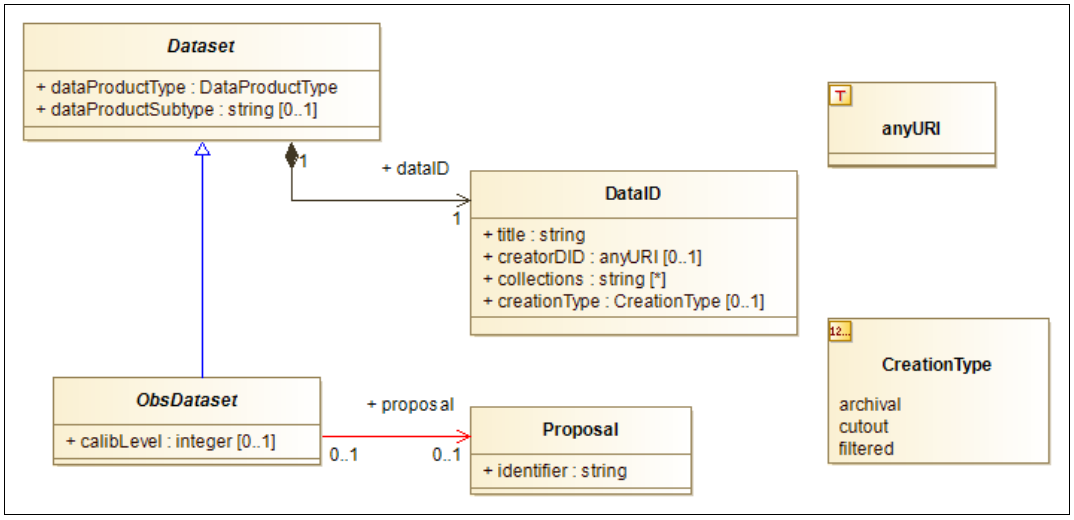
type: real **type-detail: Section 5.1**
multiplicity: 1

Decimal number representing the elapsed time.

Appendix A: Modeling Conventions

1 Diagram notation

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



1.1 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.

1.2 DataType

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

1.3 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.

1.4 Generalization

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

1.5 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.

1.6 Reference

The reference relation is indicated by a red 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.

1.7 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).

2 Model Identification metadata

Interoperability of datasets requires that there be a standardized method for identifying the specific type of dataset, and which model(s) and versions thereof it conforms to. These elements are not properties of the dataset, but rather, of the Model itself. We provide this information via stereotypes assigned to the model packages (e.g. Dataset, Char, STC, IVOA).

2.1 Model stereotype

The Model stereotype (<<model>>) consists of a set of Model properties which identify a particular model and its dependencies. Each model should specifically state the appropriate values for these properties.

2.1.1 name:string[1]

The model name. The value must match the name of the model package itself. This string identifies the particular model type (eg. Dataset, Char, STC).

2.1.2 version:string[1]

The version of this model. To be represented as a string with format "<version>.<subversion>"

2.1.3 prefix:string[1]

Sometimes referred to as 'namespace', the prefix is a tag which is used to label elements of a particular model. Each model must declare a prefix string which is unique within the IVOA to tag

elements from those models. A typical use of the prefix is in the construction of element UTYPE strings.

2.1.4 url:anyURI[1]

A URL from which the full model description may be obtained (e.g. XML schema).

2.1.5 imports:Import[*]

Here, we specify which the models on which this model is dependent. This model uses and/or extends elements from the Characterisation and STC Data models. In this document, we provide descriptions and supporting information about usage of these objects in a particular context. The originating documents, however, remain the definitive source for element definitions.

2.2 Import Stereotype

The <<import>> stereotype is attached to Packages representing imported models. It identifies the model by name, and provides URLs from which the full description may be obtained.

2.2.1 name:string[1]

The name of the imported model. This name MUST match the 'name' property of imported model's Model metadata.

2.2.2 version:string[1]

The version of this model. To be represented as a string with format "<version>.<subversion>"

2.2.3 url:anyURI[1]

A URL from which the full model description may be obtained (e.g. XML schema).

3 Extensibility

There is no formal mechanism in the IVOA defining how users may extend models with their own content. However, the above Model identification metadata provides a simple means to do so. Using this process, a user would model their content as an extension of the IVOA standard.

3.1 Model

3.1.1 name

The user-defined model would need a name unique from that of the standard.

3.1.2 prefix

A unique prefix must be defined for the user-defined model elements. Users must take care not to make use of prefix tags which are associated with current IVOA standards, (e.g. ’cha’, ’spec’, ’ssa’, ’stc’). At the time of this writing, there is no central repository of reserved namespace strings.

3.1.3 imports

The user defined model should declare the IVOA standard being extended as an imported model. Fields for the imported model name and url may be obtained from that standard's documentation.

3.2 Scope

We permit any object modeled in this document to be extended with user-defined content, with the following restrictions:

- Follow VO-DML modeling practices.
- Values of extended content must be consistent with the content of modeled data. That is, using the IVOA base primitive types, Quantity, and STC Coordinates as appropriate.
- Since extended content, by definition, does not follow the corresponding model, it is not possible for general applications to interpret complex structures within that content. It is, therefore, recommended that users define extended content in such a way as to avoid ambiguity between its components.

3.3 Support

Applications should, but are not required to, provide the following support for extended content:

- Retain existence of extended content, including namespace and UTypes.
- Retain association with modeled component.
- Provide access to extended content by users.

Appendix C: Dataset Metadata Model Summary

Dataset Model Identification				
Model Element	**Datatype**	**Mult.**	**Meaning**	**value**
Model identification				
Model				
Model.name	string	1	Data model name and version	"Dataset Metadata"
Model.version	string	1	Data model version	"1.0"
Model.prefix	string	1	Data model prefix tag	"ds"
Model.url	anyURI	1	Reference URL for model	<TBD>
Imported Model				
Import.name	string	1	Imported model name	"Char"
Import.version	string	1	Imported model version	"1.13"
Import.url	anyURI	1	Reference URL for imported model	<TBD>
Imported Model				
Import.name	string	1	Imported model name	"STC"
Import.version	string	1	Imported model version	"2.0"
Import.url	anyURI	1	Reference URL for imported model	<TBD>
Imported Model				
Import.name	string	1	Imported model name	"ivoa"
Import.version	string	1	Imported model version	"1.0"
Import.url	anyURI	1	Reference URL for imported model	<TBD>

Dataset Model Summary				
Model Element	**Datatype**	**Mult.**	**Meaning**	**UCD1+**
Dataset Model Elements				
Characterisation	Characterisation		Direct extension of Chararcterisation:Char	
Contact	Contact			
Contact.email	string	0..1	Contact email	meta.ref.url;meta.email
Contact.name	string	1	Contact name	meta.bib.author;meta.curation
Curation	Curation			
Curation.contact	Contact	0..1		
Curation.publisher	string	1	Publisher	meta.curation

Dataset Model Summary				
Model Element	**Datatype**	**Mult.**	**Meaning**	**UCD1+**
Curation.publisherDID	anyURI	0..1	Publisher specified dataset ID	meta.ref.url;meta.curation
Curation.publisherID	anyURI	0..1	URI for VO Publisher	meta.ref.url;meta.curation
Curation.reference	string	0..*	URL or Bibcode for documentation	meta.bib.bibcode
Curation.releaseDate	datetime	0..1	Date curated dataset last modified	time.release
Curation.rights	RightsType	0..1	Proprietary restrictions level	meta.code
Curation.version	string	0..1	Publisher version of the dataset	meta.version;meta.curation
DataID	DataID			
DataID.collection	string	0..*	Collection name(s)	meta.id
DataID.contributor	string	0..*	Contributor(s)	
DataID.creationType	CreationType	0..1	Dataset creation type	
DataID.creator	string	0..1	VO Creator ID	meta.curation
DataID.creatorDID	anyURI	0..1	Creator defined Dataset Identifier	meta.id
DataID.datasetID	anyURI	0..1	IVOA Dataset Identifier	meta.id;meta.dataset
DataID.date	datetime	0..1	Data processing/creation date	time.epoch;meta.dataset
DataID.logo	anyURI	0..1	URL for creator logo	meta.ref.url
DataID.observationID	string	0..1	Observation ID	meta.id
DataID.title	string	1	Dataset title	meta.title;meta.dataset
DataID.version	string	0..1	Version of dataset	meta.version;meta.dataset
Dataset				
Dataset.curation	Curation	1	Dataset curation metadata	
Dataset.dataID	DataID	1	Dataset identification metadata	
Dataset.dataProductSubType	string	0..1	Dataset subtype	meta.id
Dataset.dataProductType	DataProductType	1	Dataset or segment type	meta.id
Derived	Derived			
Derived.derivedElement	DerivedElement	0..*	property derived from analysis of data content	
DerivedScalar	DerivedScalar			
DerivedScalar.name	string	1	name of derived property	
DerivedScalar.value	Quantity	1	value of derived property	
ObsDataset	ObsDataset			
ObsDataset.calibLevel	integer	0..1	Calibration level	meta.code;obs.calib
ObsDataset.characterisation	Characterisation	1		
ObsDataset.coordSys	AstroCoordSystem	0..*	Global coordinate systems	
ObsDataset.derived	Derived	0..1	Derived metadata	
ObsDataset.obsConfig	ObsConfig	0..1	Observatino configuration	
ObsDataset.proposal	Proposal	0..1	Proposal information	
ObsDataset.target	BaseTarget	1		

Dataset Model Summary				
Model Element	**Datatype**	**Mult.**	**Meaning**	**UCD1+**
Observation Model Elements				
AstroTarget	AstroTarget		Astronomical target	
AstroTarget.objectClass	string	0..1	Target or object class	src.class
AstroTarget.redshift	real	0..1	Target redshift	src.redshift
AstroTarget.spectralClass	string	0..1	Object spectral class	src.spType
AstroTarget.VarAmpl	real	0..1	Target variability amplitude - typical	src.var.amplitude
Bandpass	Bandpass		Direct extension of ObservingElement	
Bandpass.name	string	1	Band	instr.bandpass
BaseTarget	BaseTarget			
BaseTarget.description	string	0..1	Target descriptive text	meta.note;src
BaseTarget.name	string	1	Target name	meta.id;src
BaseTarget.position	Position	0..1	Target location (eg: RA, DEC)	pos[.eq];src
DataSource	DataSource		Direct extension of ObservingElement	
DataSource.name	string	1	Original data type	
Facility	Facility		Direct extension of ObservingElement	
Facility.name	string	1	Facility name	meta.id;instr.tel
Instrument	Instrument		Direct extension of ObservingElement	
Instrument.name	string	1	Instrument ID	meta.id;instr
ObsConfig	ObsConfig			
ObsConfig.observingElement	ObservingElement	0..*	Observation configuration parameter	
Observation	Observation			
Observation.obsConfig	ObsConfig	1	Observation configuration metadata	
Observation.observationID	string	1	Observation ID	
Observation.proposal	Proposal	0..1	Proposal which spawned the observation	
Observation.result	ObsDataset	0..*	Dataset(s) resulting from the observation	
Observation.target	BaseTarget	1	Target or goal of the observation	
ObservingElement	ObservingElement			
ObservingElement.name	string	1	Identifies the specific instance of the element	
Proposal	Proposal			
Proposal.Identifier	string	1	Proposal ID	meta.id;obs.proposal
Target	Target		Generic Target	
Target.objectClass	string	0..1	Target or object class	src.class

OPEN QUESTION
QualityCode

The entity is represented by a numerical code.

3.3.1.1 QualityCode.codeNum:integer

Numerical code.

3.3.1.2 QualityCode.definition:string

String definition of the code.

References

[1] *Resource Metadata for the Virtual Observatory*: Version 1.12, 02 March 2007
http://www.ivoa.net/Documents/latest/RM.html
[2] ”IVOA Identifiers”: Version 1.12, 14 March 2007
http://www.ivoa.net/Documents/latest/IDs.html
[3] ”Data Model for Astronomical DataSet Characterisation”: Version 1.13
http://www.ivoa.net/Documents/REC/DM/CharacterisationDM-20080325.pdf
[4] ”ST-ECF newsletter, issue #42”, June 2007
http://www.spacetelescope.org/about/further information/newsletters/html/newsletter42.html
[5] ”DER SNR: A simple and general spectroscopic signal-to-noise measurement algorithm”;
http://www.stecf.org/software/ASTROsoft/DER SNR
[6] ”Simulation Data Model”: Version 1.0, 03 May 2012;
http://www.ivoa.net/documents/SimDM/20120503/REC-SimulationDataModel-1.00-20120503.pdf
[7] ”An encoding system to represent stellar spectral classes in archival databases and catalogs": Version 1.04, 2011 Dec 15
http://www.ivoa.net/documents/Notes/SpectClasses/20111215/SpectClasses-20111215.pdf
[8] ”VO-DML a consistent modeling language for IVOA data models”: Version 1.00-20150206
http://volute.googlecode.com/svn/trunk/projects/dm/vo-dml/doc/VO-DML-WD-v1.0.pdf
[9] ”Units in the VO”: Version 1.0
http://www.ivoa.net/Documents/VOUnits/20120820/PR-VOUnits-1.0-20120820.pdf
[10] ”The UCD1+ controlled vocabulary”: Version 1.23
http://www.ivoa.net/Documents/REC/UCD/UCDlist-20070402.pdf
http://cdsweb.u-strasbg.fr/UCD/ucd1p-words.txt
[11] ”An IVOA Standard for Unified Content Descriptors”: Version 1.10
http://www.ivoa.net/Documents/REC/UCD/UCD-20050812.pdf
http://www.ivoa.net/Documents/latest/UCD.html