

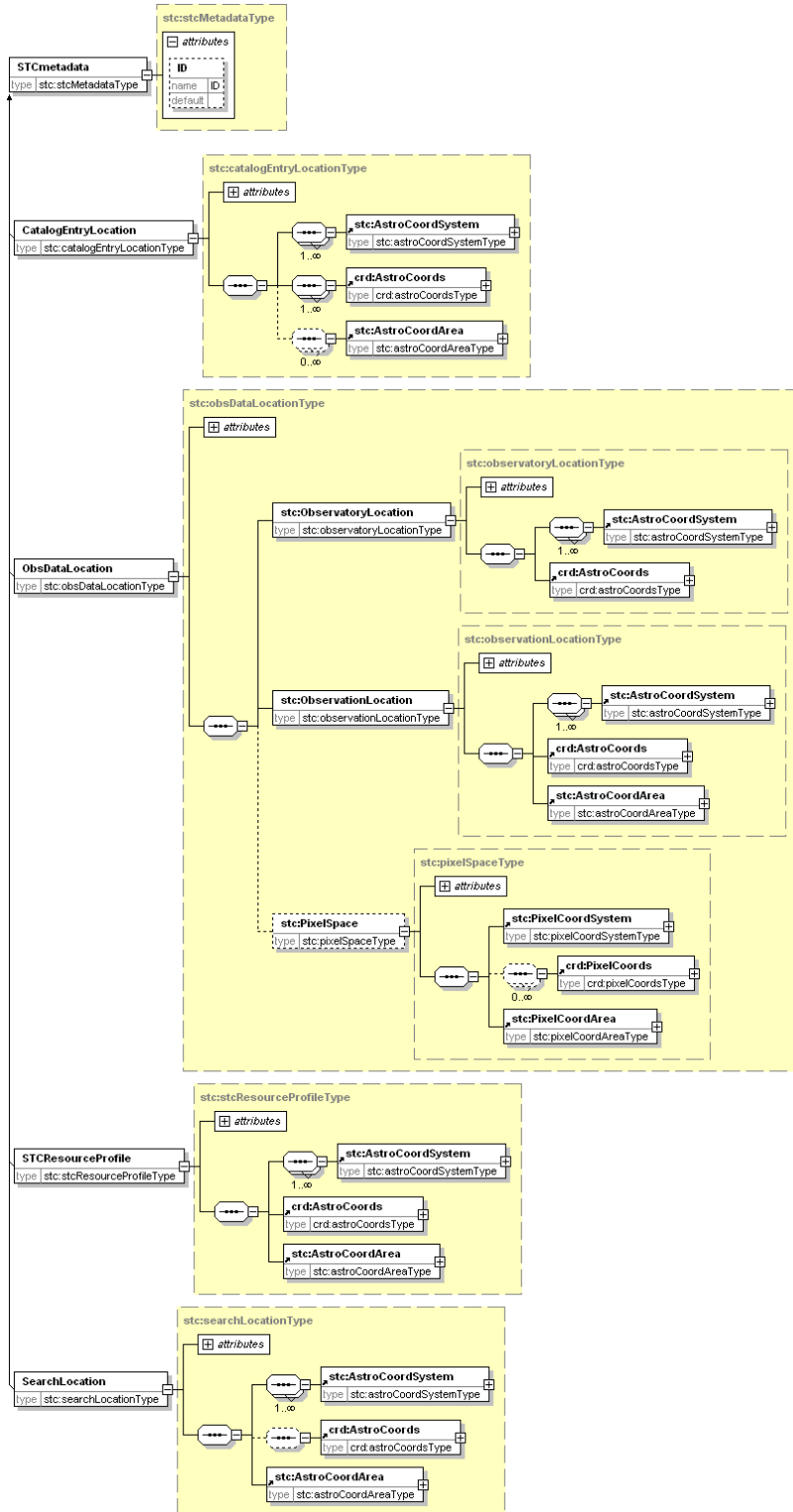
## Space-Time Coordinate (STC) Metadata XML Implementation (in XMLSpy Diagrams)

In the following pages the XML implementation of STC is illustrated through the UML-like diagrams created by XMLSpy.

Toplevel Elements	2
STCmetadata	2
Coordinate Systems	3
CoordSys	3
AstroCoordSystem	4
PixelCoordSystem	4
Coordinate Frames	5
TimeFrame	5
SpaceFrame	6
SpectralFrame	6
RedshiftFrame	7
GenericCoordFrame	7
PixelCoordFrame	7
Coordinate Frame Components	8
SpaceRefFrame	8
ReferencePosition	9
CoordFlavor	10
Coordinates	11
Coords	11
AstroCoords	12
Coordinate	13
ScalarCoordinate	14
AstronTime	15
Spatial Position	16
astroCoordsFileType	17
Coordinate Area	18
CoordArea	18
PixelCoordArea	18
SpatialInterval	19
Coordinate Interval	20
CoordInterval	20
Spatial Region	21
Negation, Union, Intersection	21
Allsky, Circle, Ellipse	22
Polygon, Sector	23
Convex, ConvexHull	24

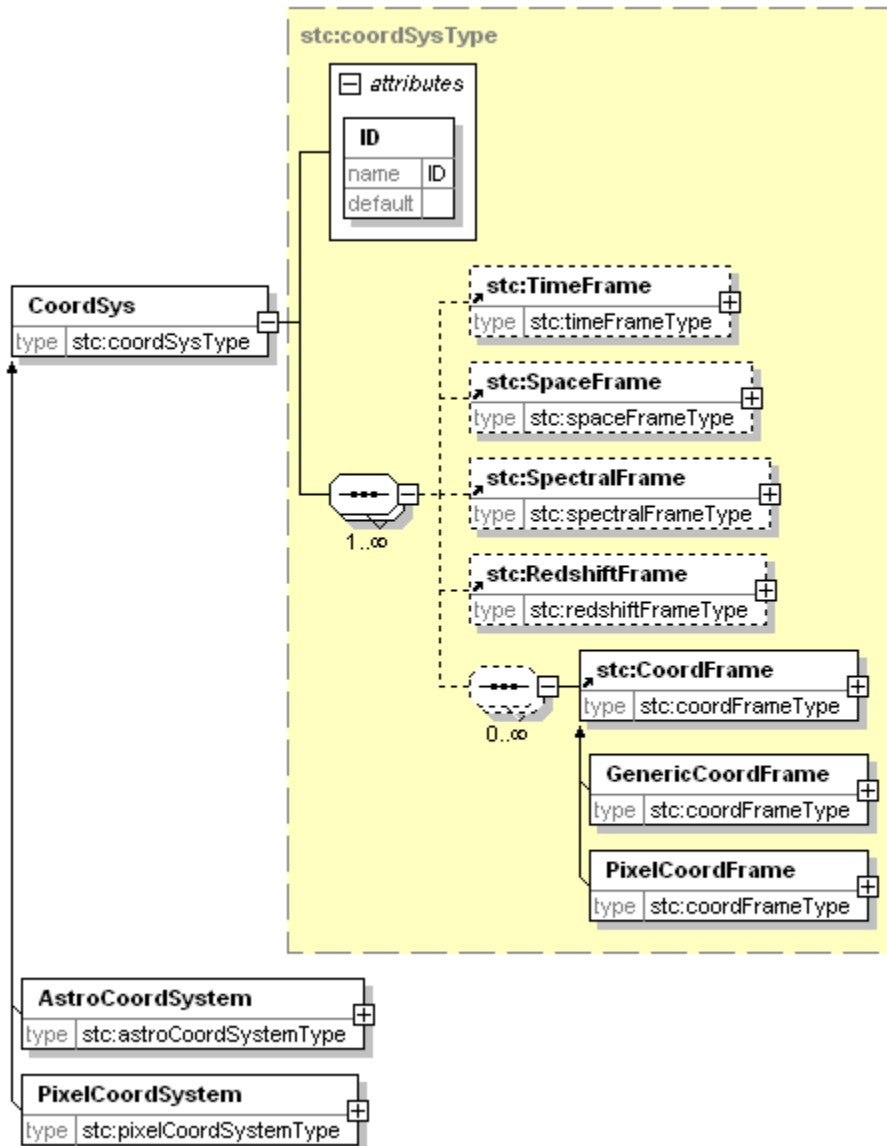
## Top-level Metadata Elements

There are currently four types of **STCmetadata** elements derived, all based on the `stcDescriptionType` consisting of three elements: `CoordSys`, `Coords`, and `CoordArea`.



## Coordinate Systems

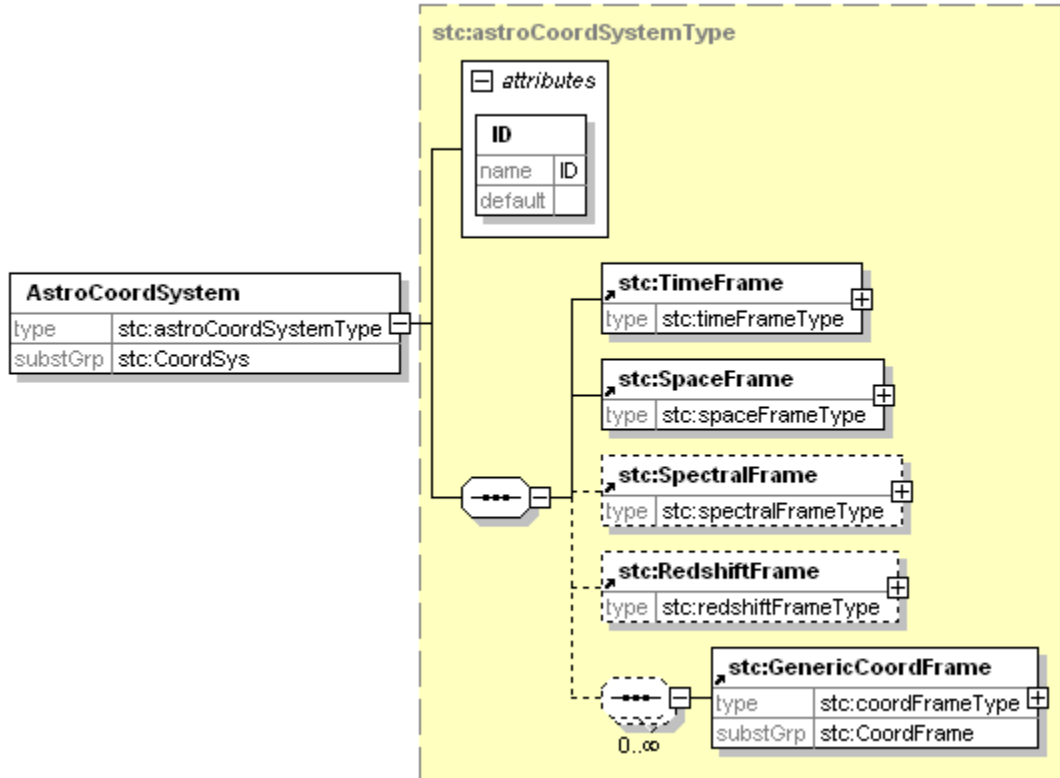
Basic **CoordSys** contains one or more **CoordFrames** and is required to have an ID. There are 5 specific Frame implementations and 1 generic Frame. There are two specific coordinate systems derived from **CoordSys**: **AstroCoordSystem** and **PixelCoordSystem**.



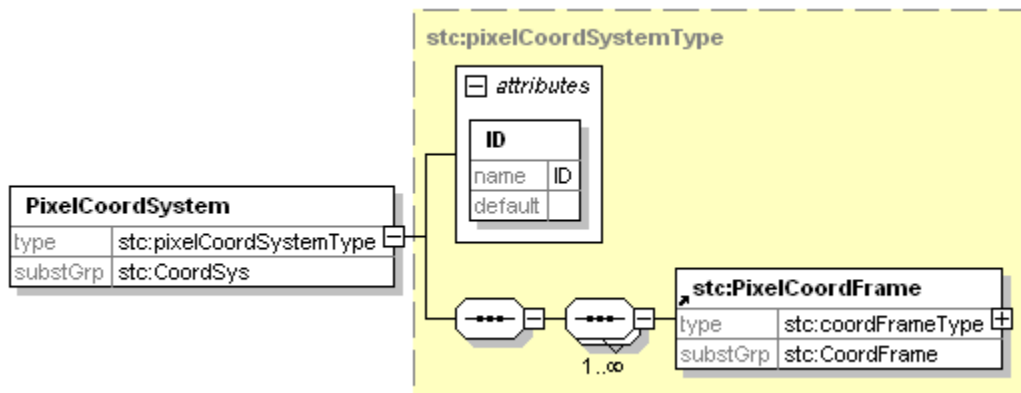
## STC XML Implementation

**AstroCoordSystem** contains 3 required Frames, an optional Frame, and may contain any number of generic Frames.

It is derived from coordSysType by restriction.

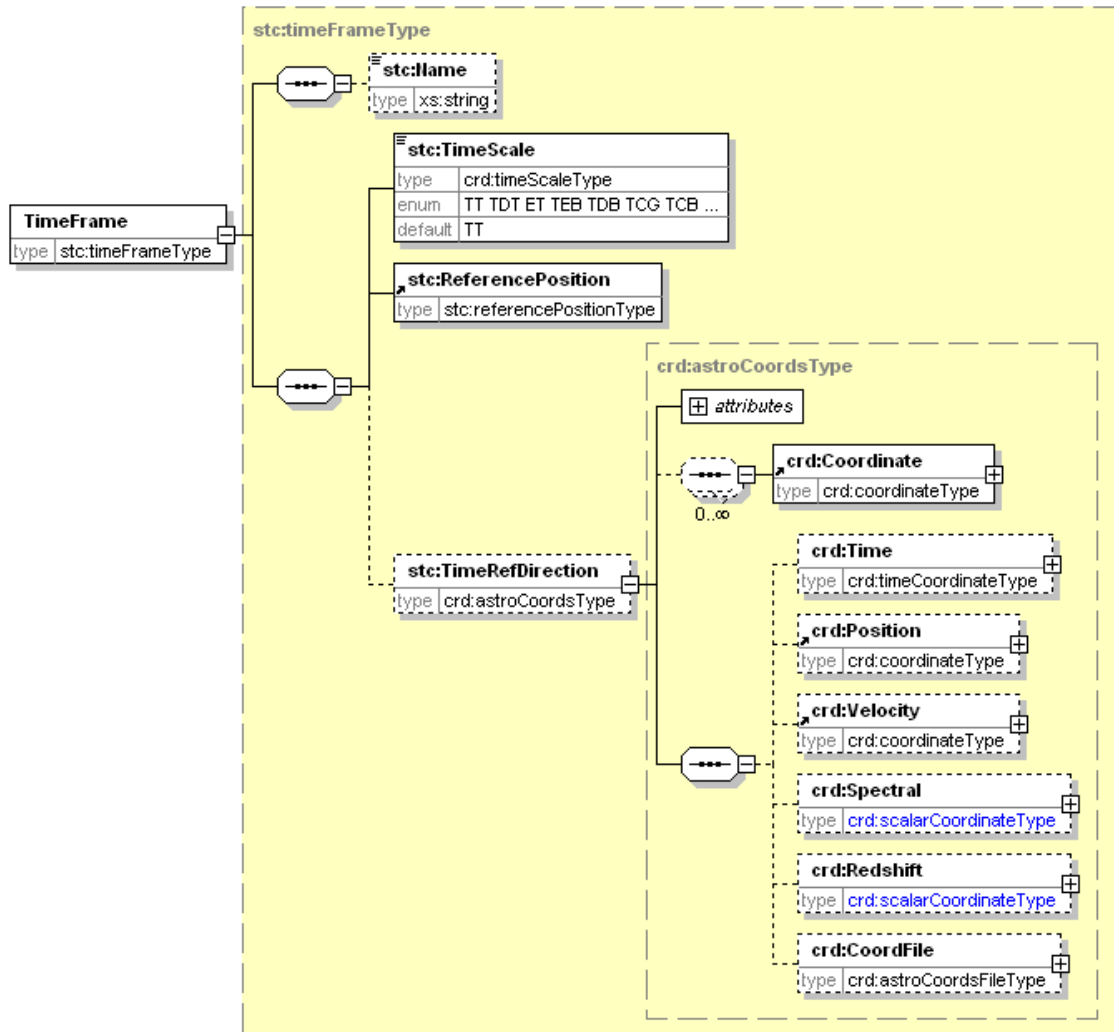


**PixelCoordSystem** is made up of Frames that only have a name



## Coordinate Frames

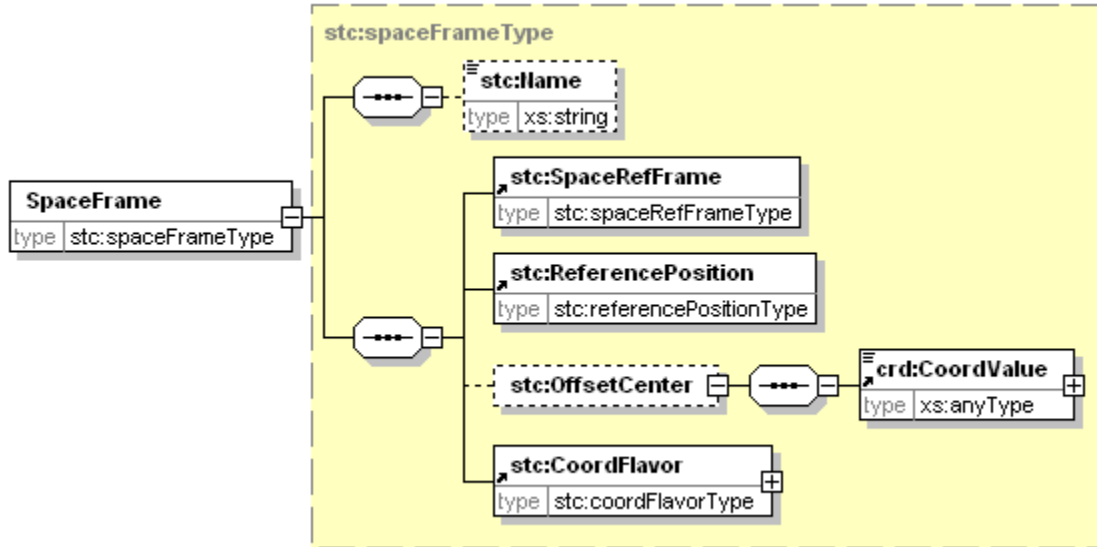
The **TimeFrame**:



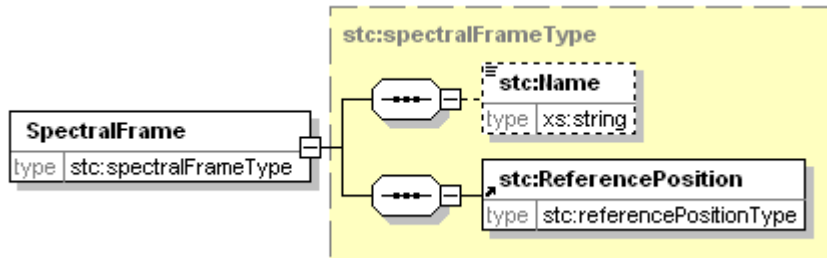
### The **SpaceFrame**

The **OffsetCenter** is added to accommodate lists with RA and Dec offsets.

Note that this is intended for pure numeric RA and Dec offsets; true angle offsets should be handled through a **CustomSpaceRefFrame**.

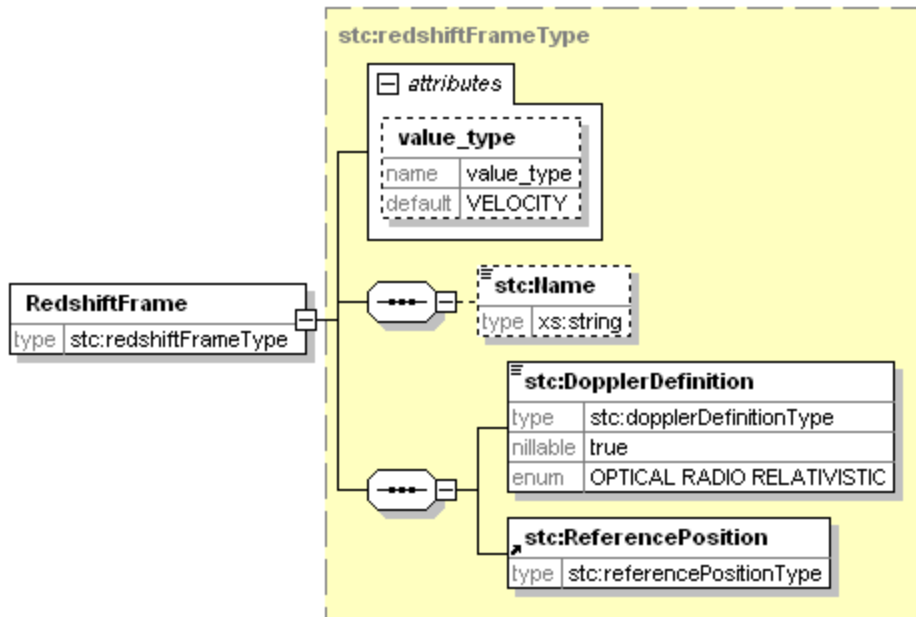


The **SpectralFrame** is simple:

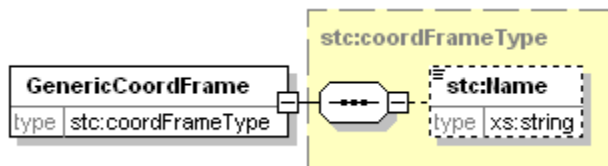


## STC XML Implementation

The **RedshiftFrame** has a little more:



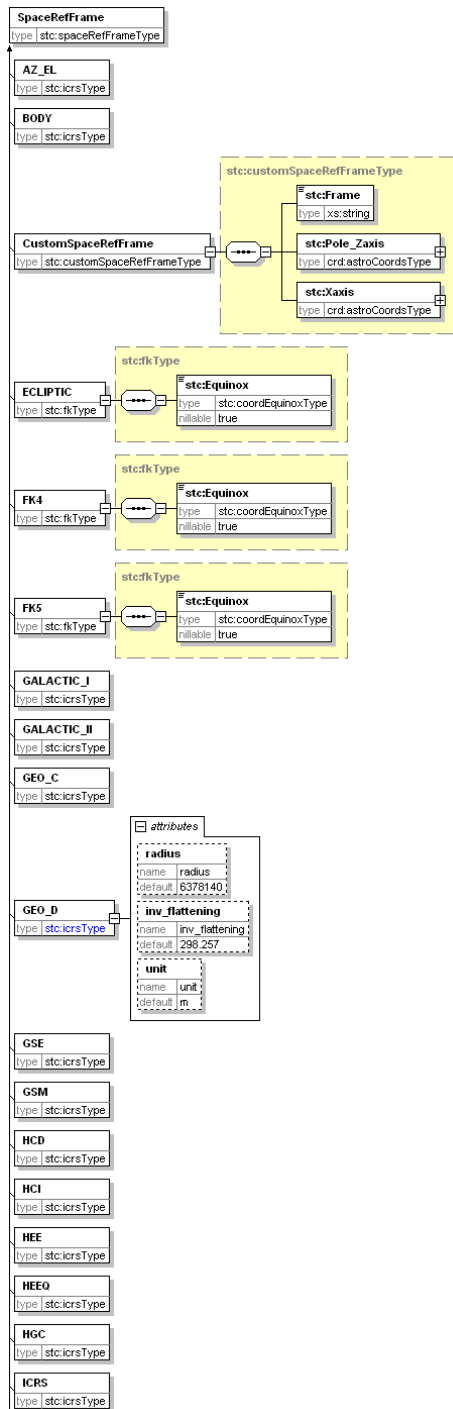
The **GenericCoordFrame** (and **PixelCoordFrame**) just have a name:



## Coordinate Frame Components

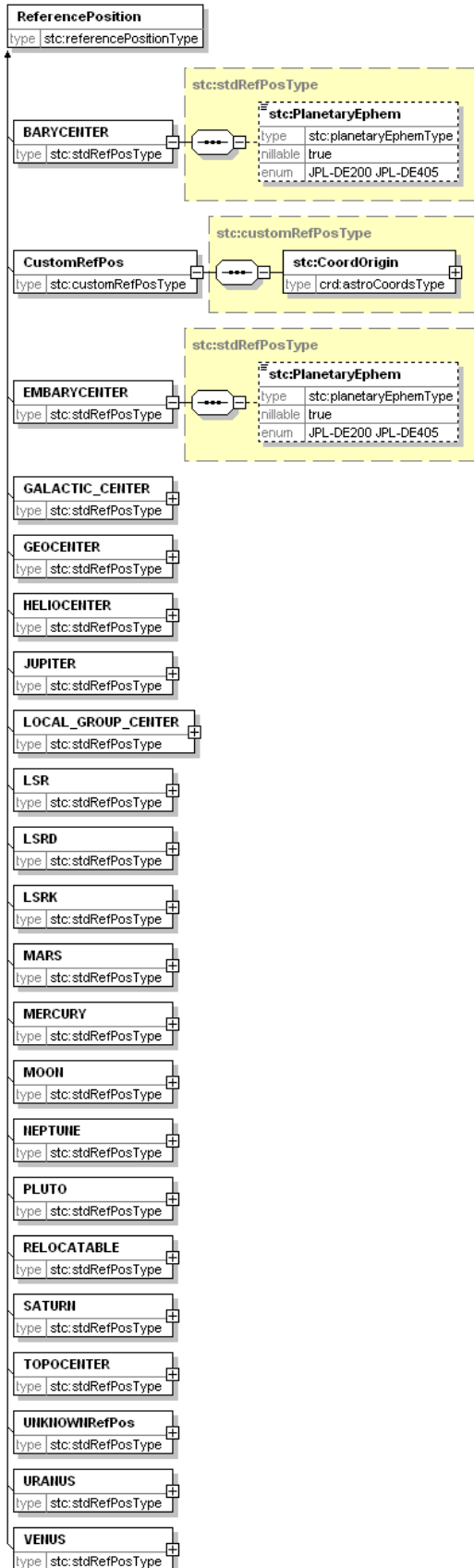
The SpaceReferenceFrame (part of SpaceFrame) uses the names of the elements in the **SpaceRefFrame** substitution group to identify the frame, rather than using a single element with enumerated values; this makes it easier to extend the list – by just adding members to the substitution group – such as solar reference systems.

Note that ICRS does not have an Equinox element. The figure below is truncated.



# STC XML Implementation

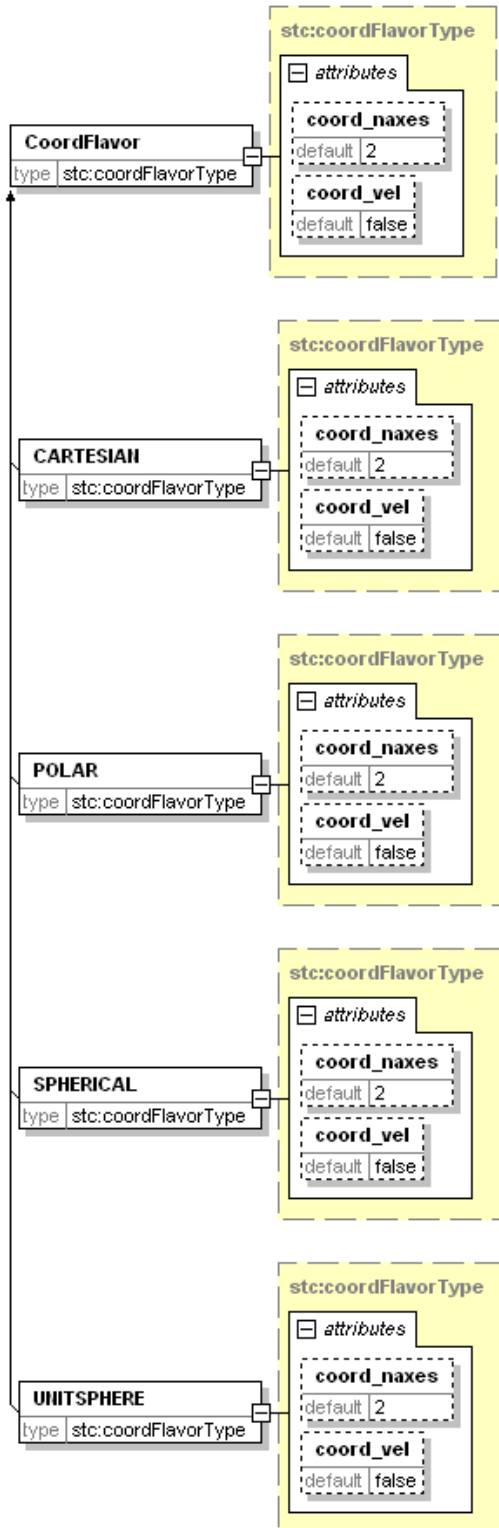
The **ReferencePosition** uses the same technique:



## STC XML Implementation

The **CoordFlavor** also determines the flavor by the names of the substitution group's elements and is similarly easily extensible.

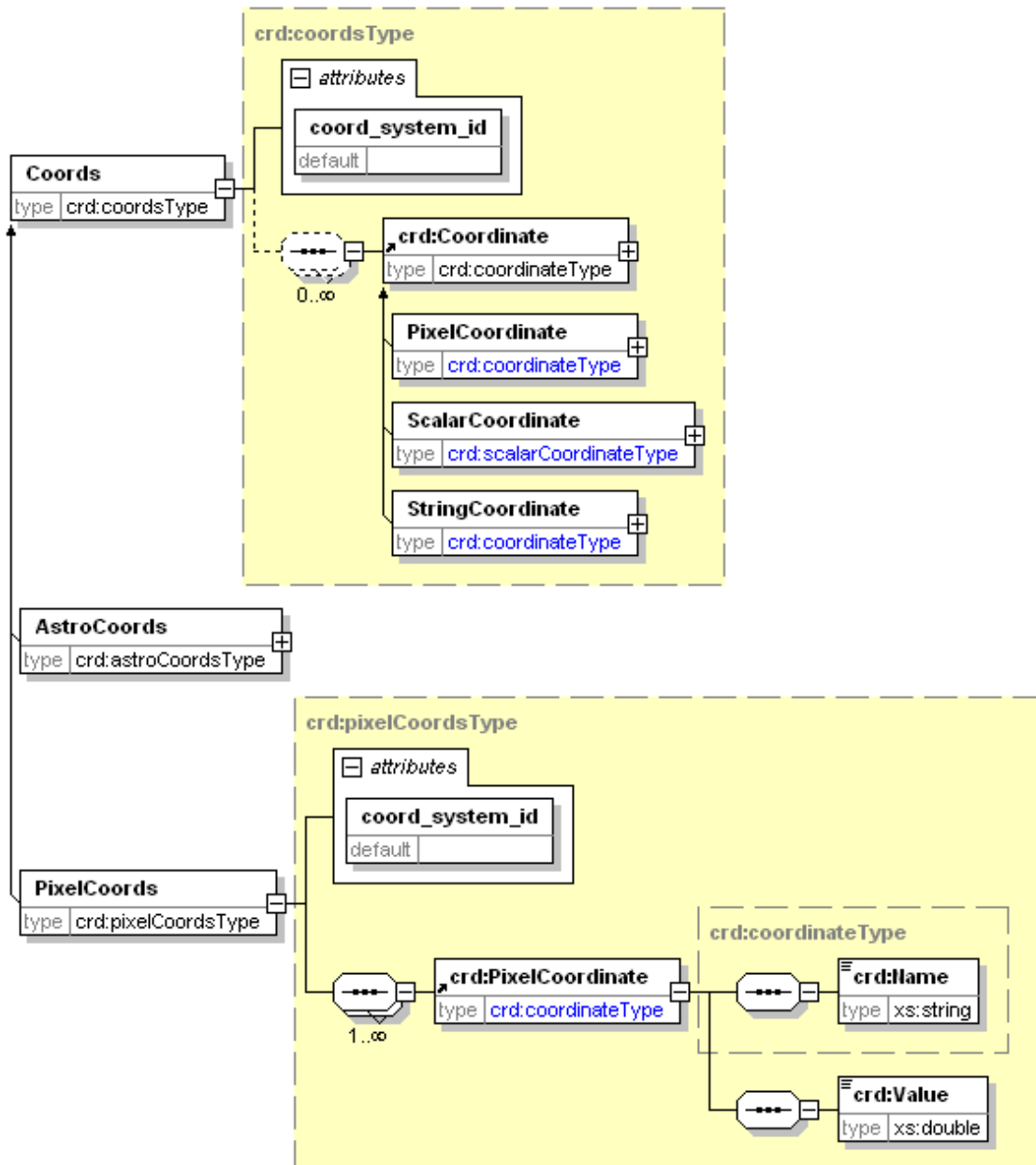
In addition, these elements have two attributes: the number of axes (`coord_naxes`, default 2) and a Boolean indicating whether velocities are present (`coord_vel`, default false)



## Coordinates

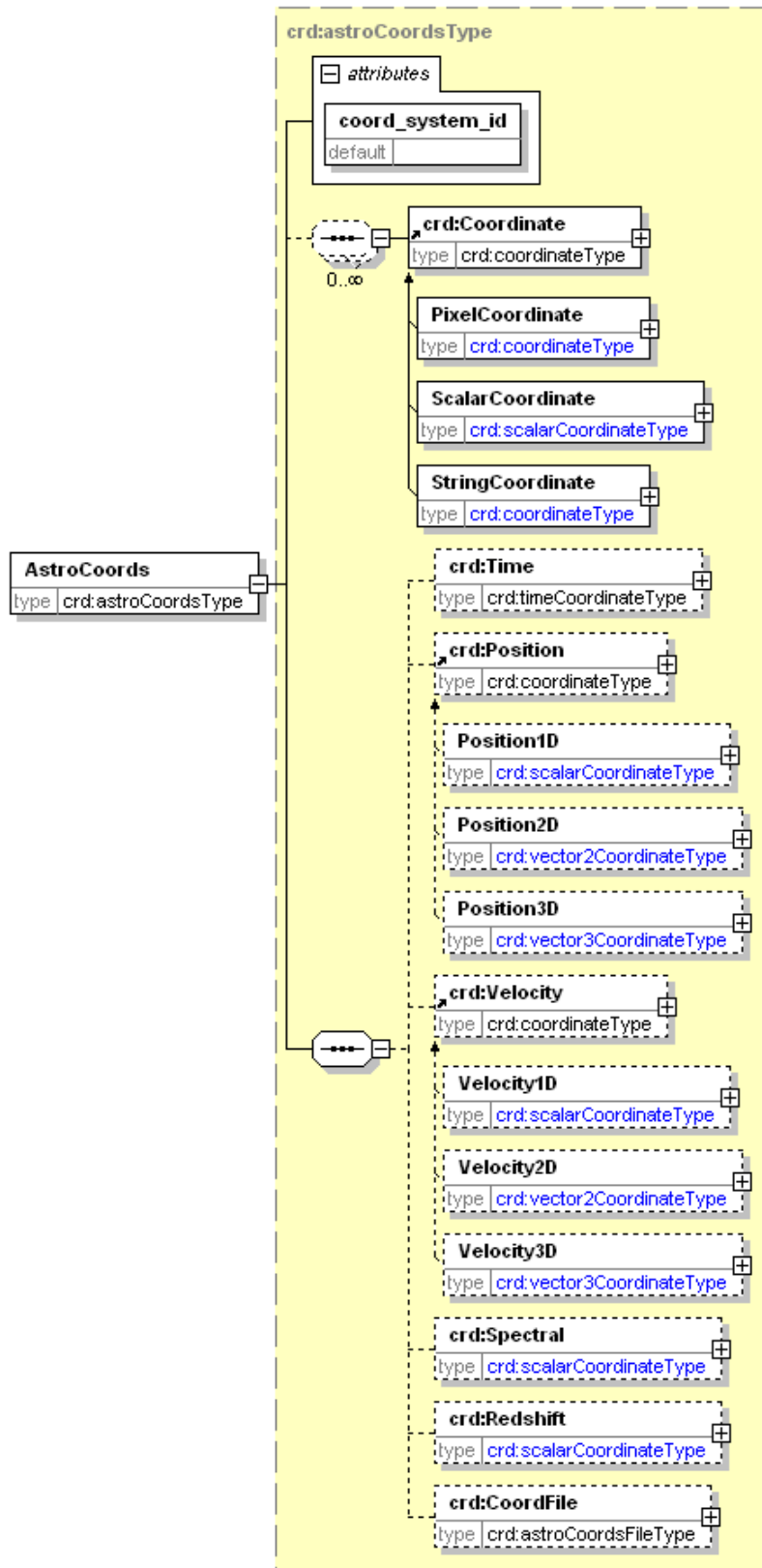
**Coords** is the generic coordinates element. It needs to refer to a specific instance of a CoordSys.

There are two derived classes: AstroCoords (by extension) and PixelCoords (by restriction).



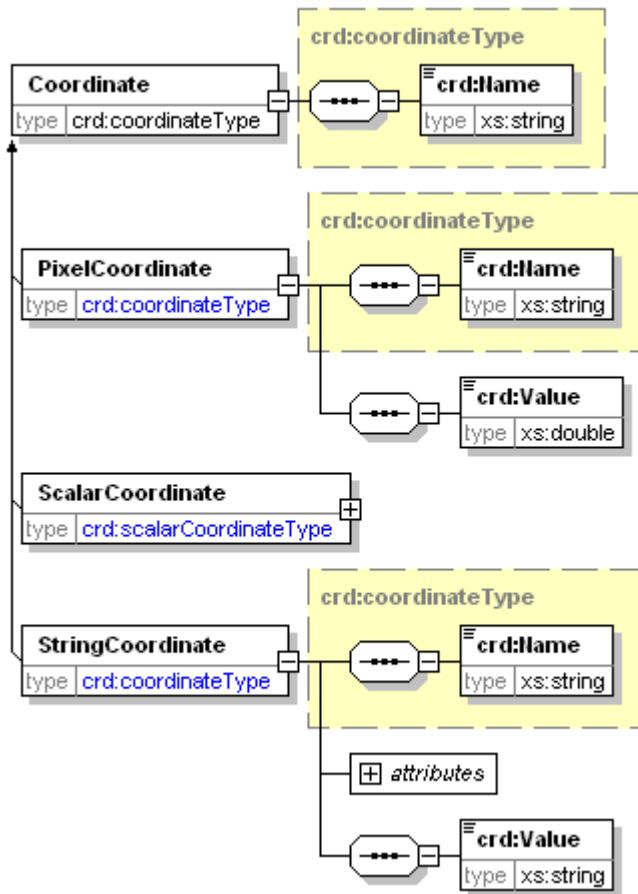
## STC XML Implementation

**AstroCoords** contains specific astronomical coordinates (time, space, spectral, redshift) and, optionally, generic coordinates



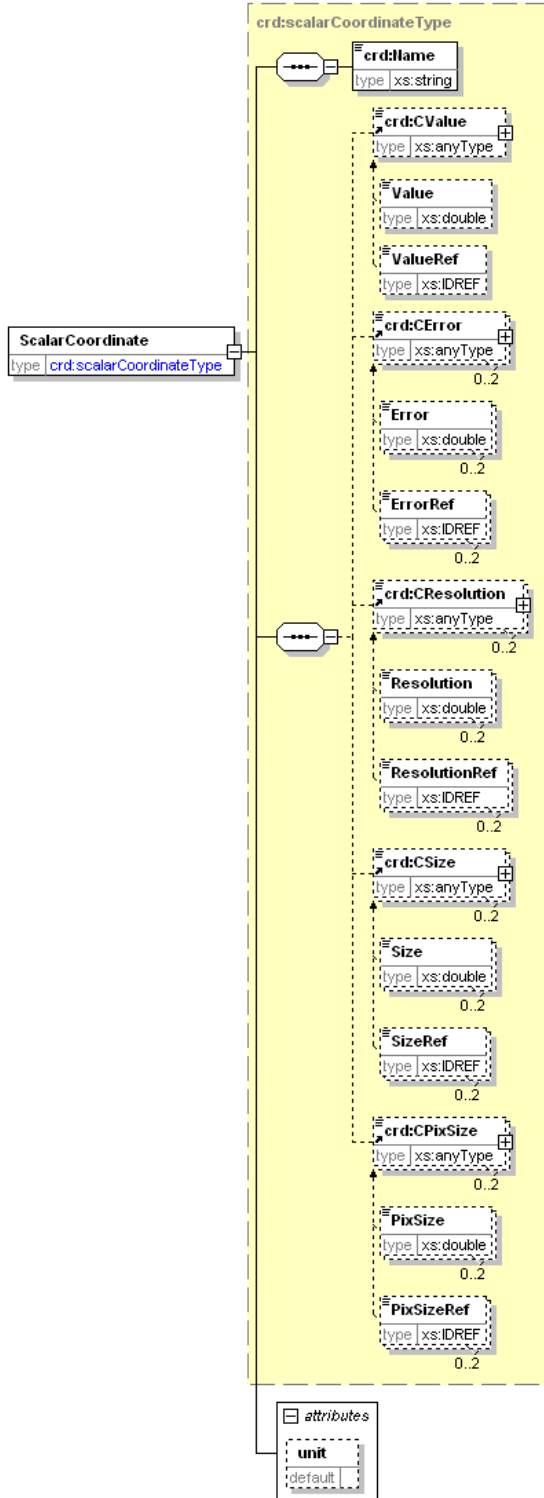
## STC XML Implementation

The **simple Coordinate** just has a Name. Its simplest derived classes add only a Value (PixelCoordinate) or a Values and an optional Unit (StringCoordinate). ScalarCoordinate is the generalized full coordinate type.

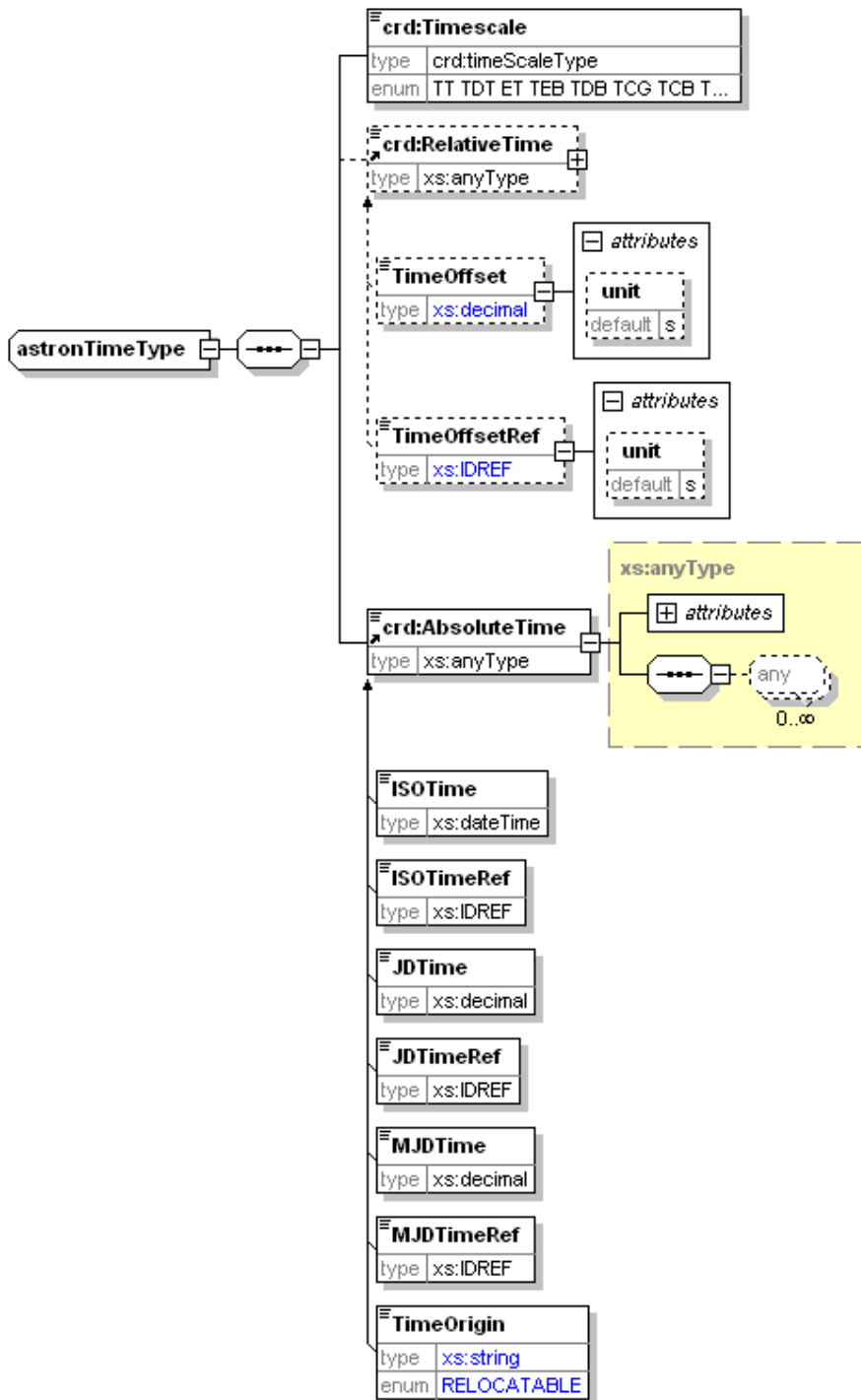


## STC XML Implementation

**ScalarCoordinate**, in addition to the Name, has an optional Unit and the full 5 coordinate components (Value, Error, Resolution, Size, and PixelSize), each of which may be an actual value or an IDREF pointing to another element in the document. Not all components need to be present, but at least one of them ought to. Note that components other than Value may appear in pairs, which would indicate a range.

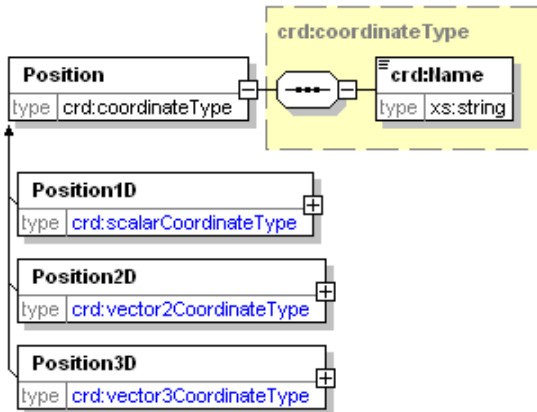


**AstronTime** is especially designed for astronomical application. It needs to contain a Timescale and an absolute time. The latter may be expressed in a subset of ISO-8601 or as Julian or Modified Julian Day. In all cases this may be an actual value or an IDREF to another element in the document. Note that (M)JD values are xs:decimal in order to achieve necessary precision. RELOCATABLE TimeOrigin s especially for simulations. In addition, time may be expressed as relative (elapsed) time in which case the AbsoluteTime element provides the reference time; in such a situation will will almost certainly want to use an IDREF for one of AbsoluteTime's derived types.

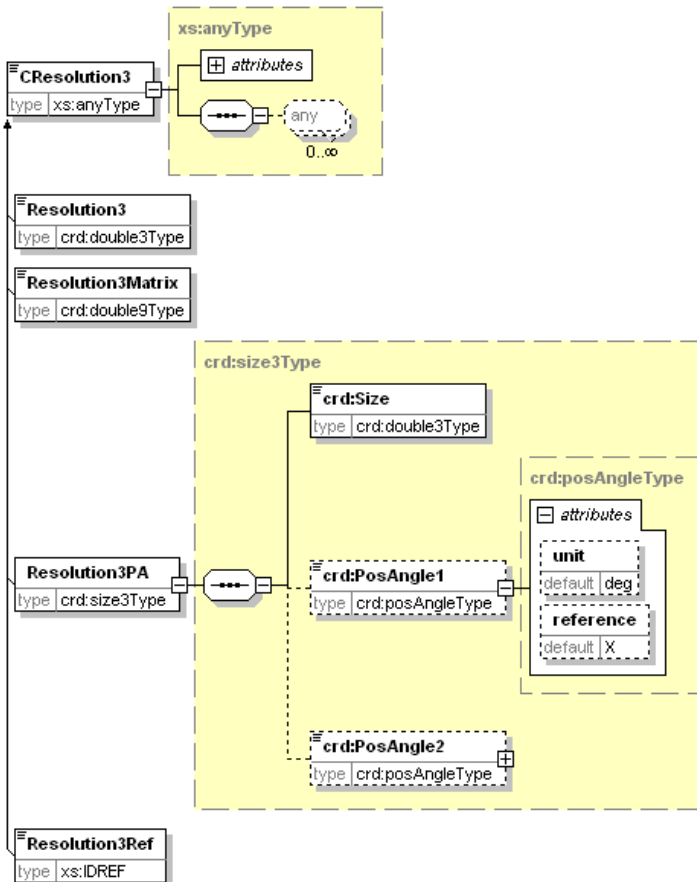


## STC XML Implementation

**Spatial Position** is more complicated Coordinate type since it refers to a compound axis; i.e., it may be 1-, 2-, or 3-dimensional.

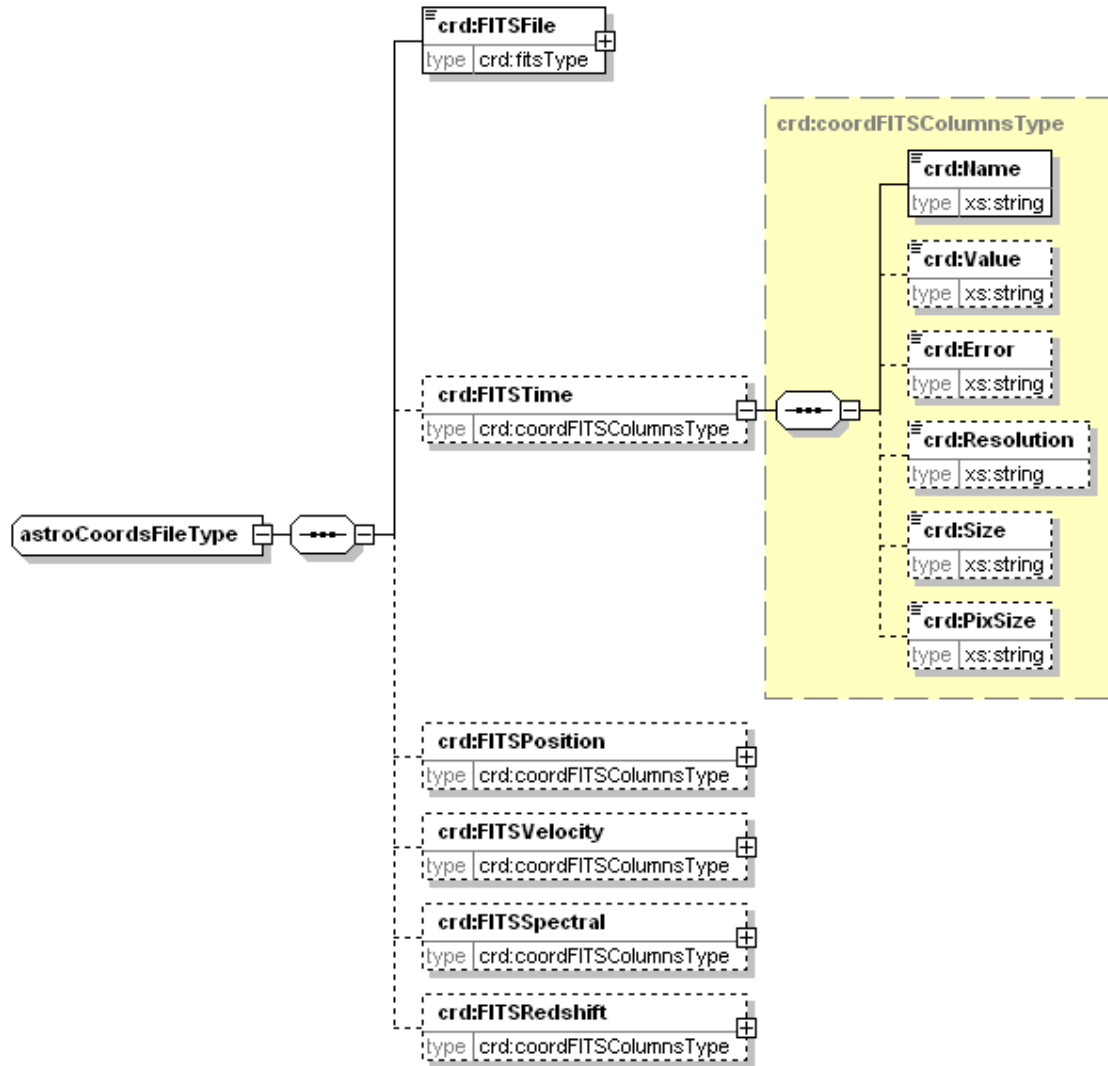


In particular, the multi-dimensional components other than Value are more complicated. Value would be a vector of 3 doubles in Position3D, but the others might include additional position angles or might be formally expressed by a 3x3 matrix. The attributes of a position angle provide its units as well as its definition.



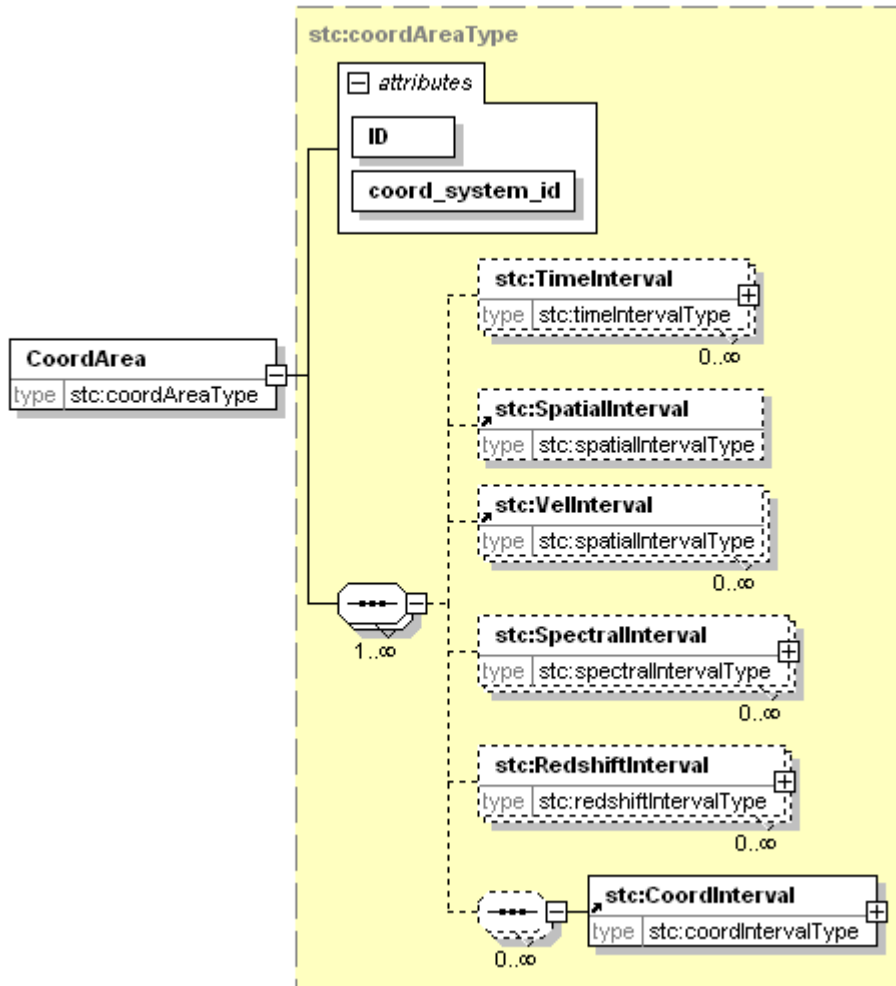
### Coordinates in a FITS file

The AstroCoords element may refer to a specific binary table HDU in a FITS file (specified through FITSFile, derived from anyURI), identifying the columns in which individual coordinate components are contained.

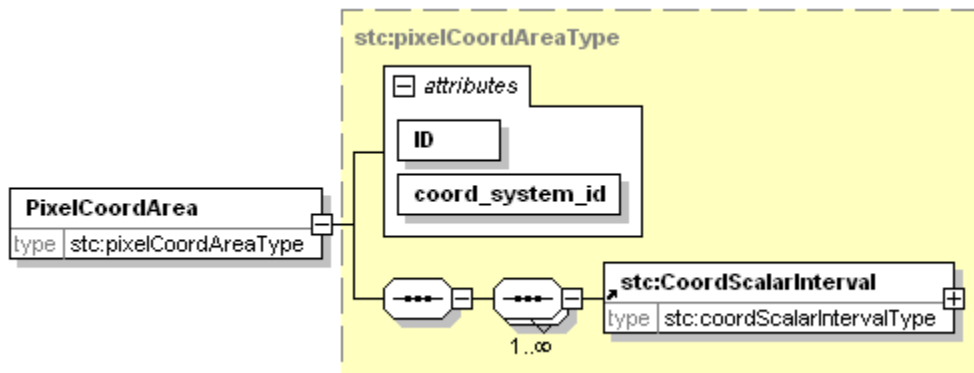


## Coordinate Area

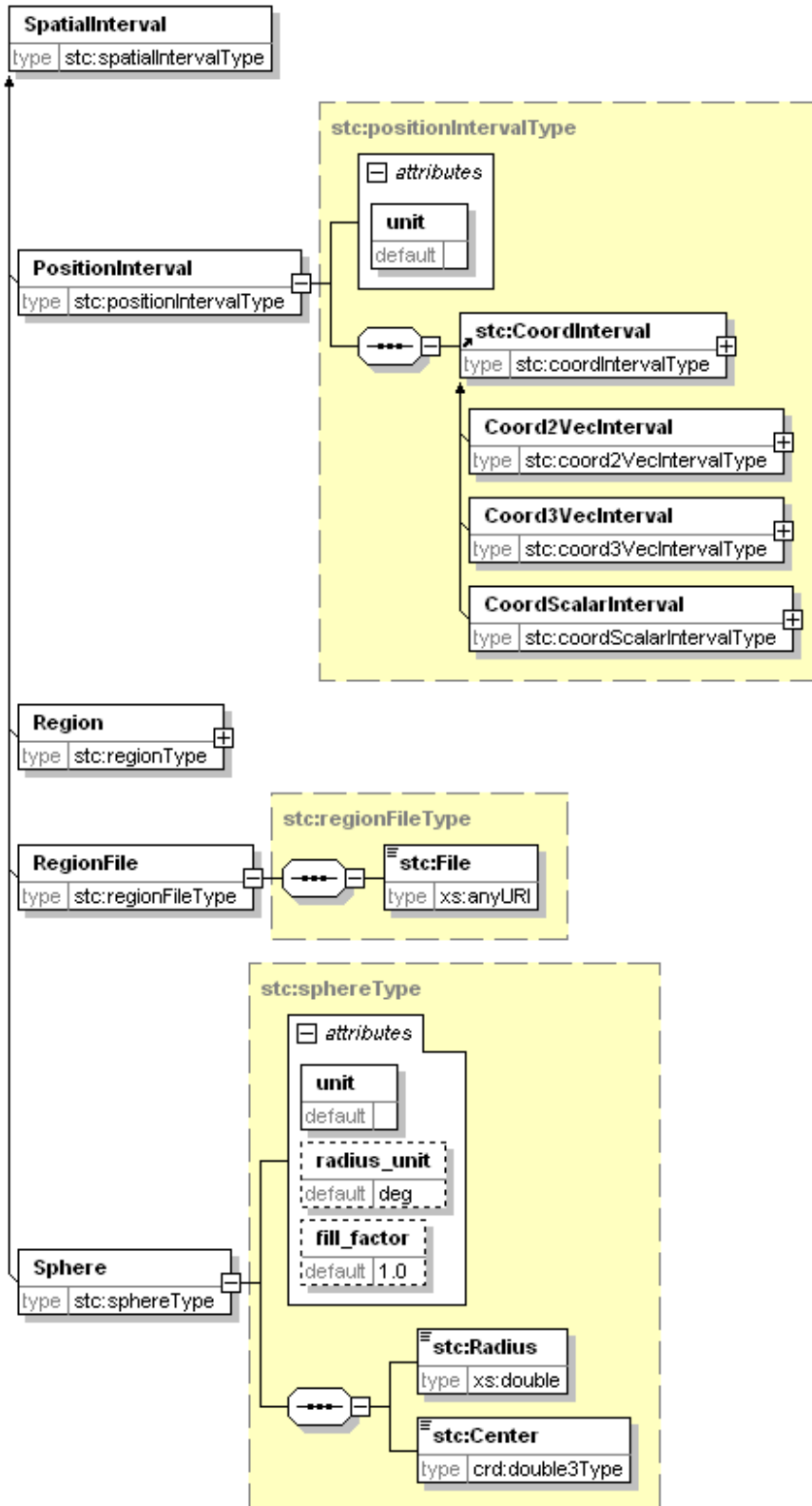
**CoordArea** specifies the volume in coordinate space that is occupied by the object to which the STC metadata is attached. It requires an IDREF pointing to a CoordSys. An area may consist of multiple intervals along each axis. Intervals have a FillFactor.



**PixelCoordArea** specifies the bounds of a pixilated data object in pixel space.

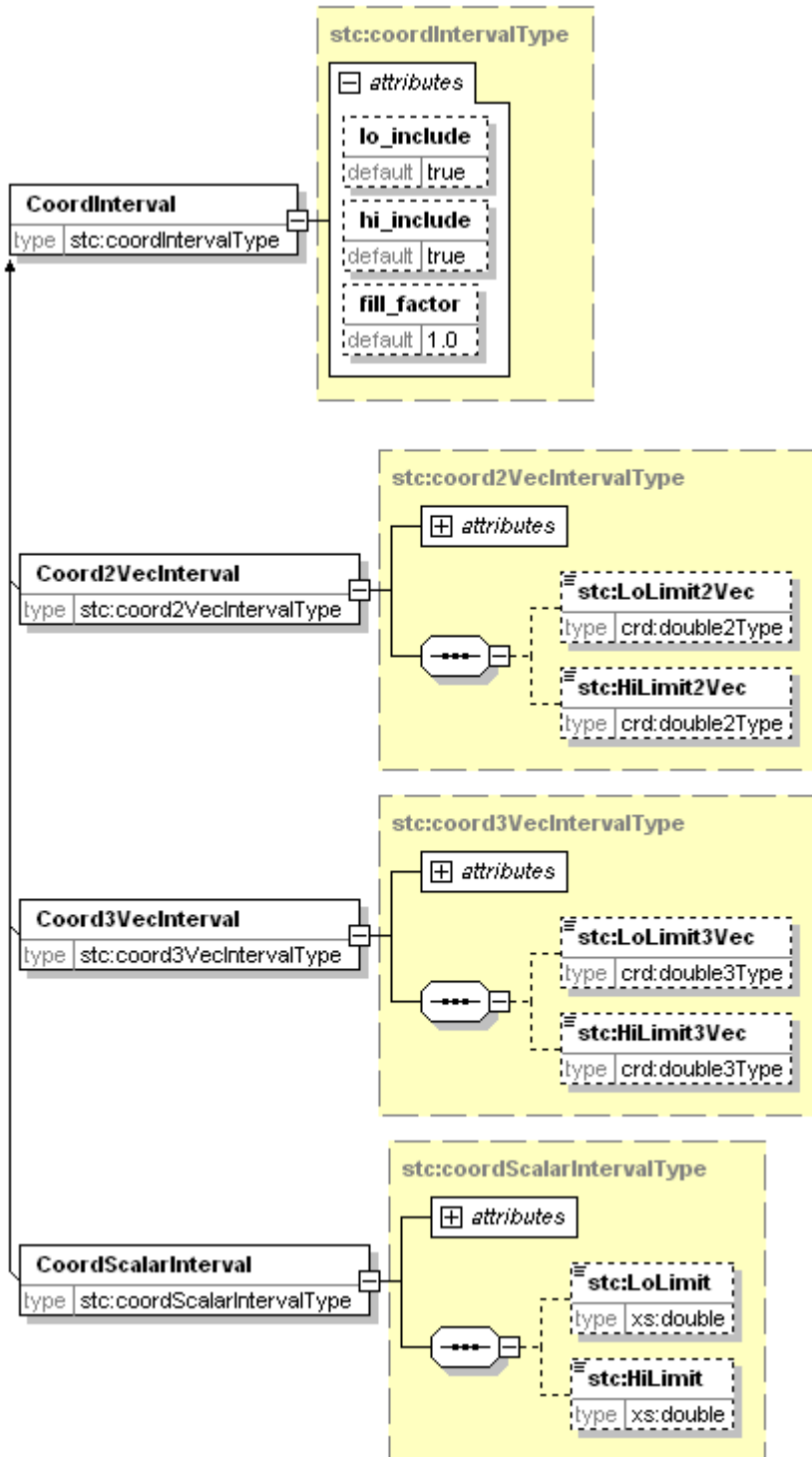


**Spatial area** is not only more complicated because it may be multi-dimensional by itself, but also because there are additional options: a Region (specified directly or in a file) or a Sphere (or Circle/Cone).



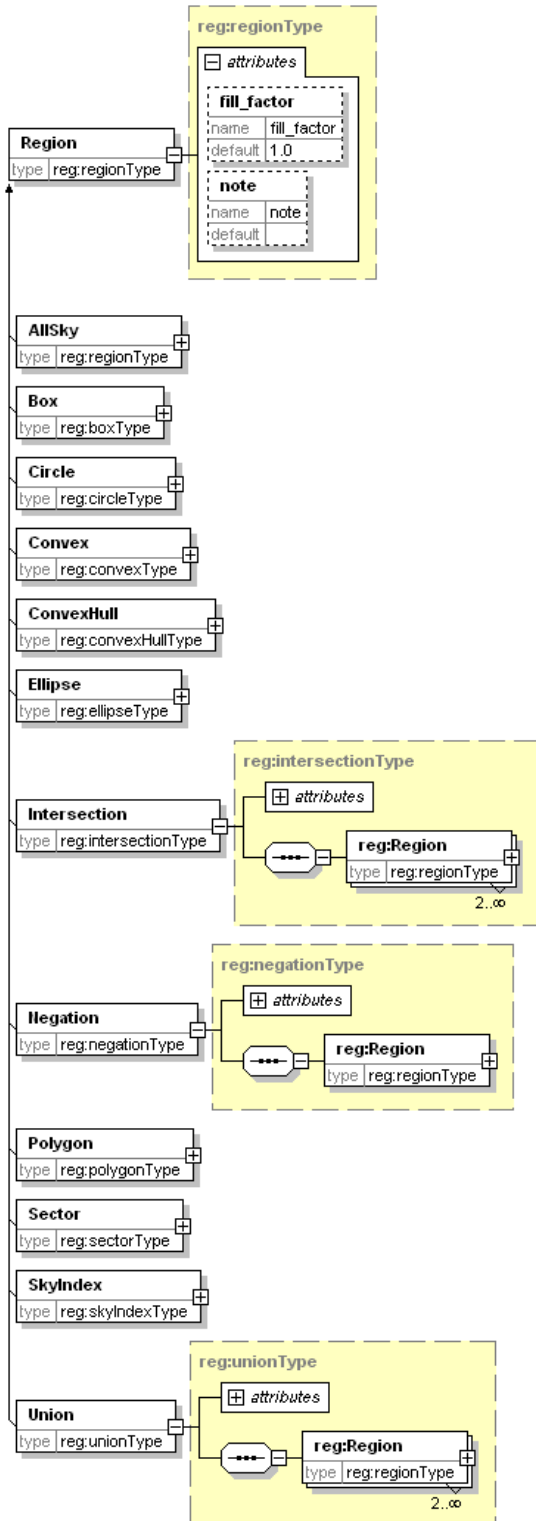
### Coordinate interval

This is the most fundamental element type to specify an area; it may be 1-, 2-, or 3-dimensional. Bounds may or may not be included. Presence of only one of the bounds indicates a lower, respectively upper, limit.



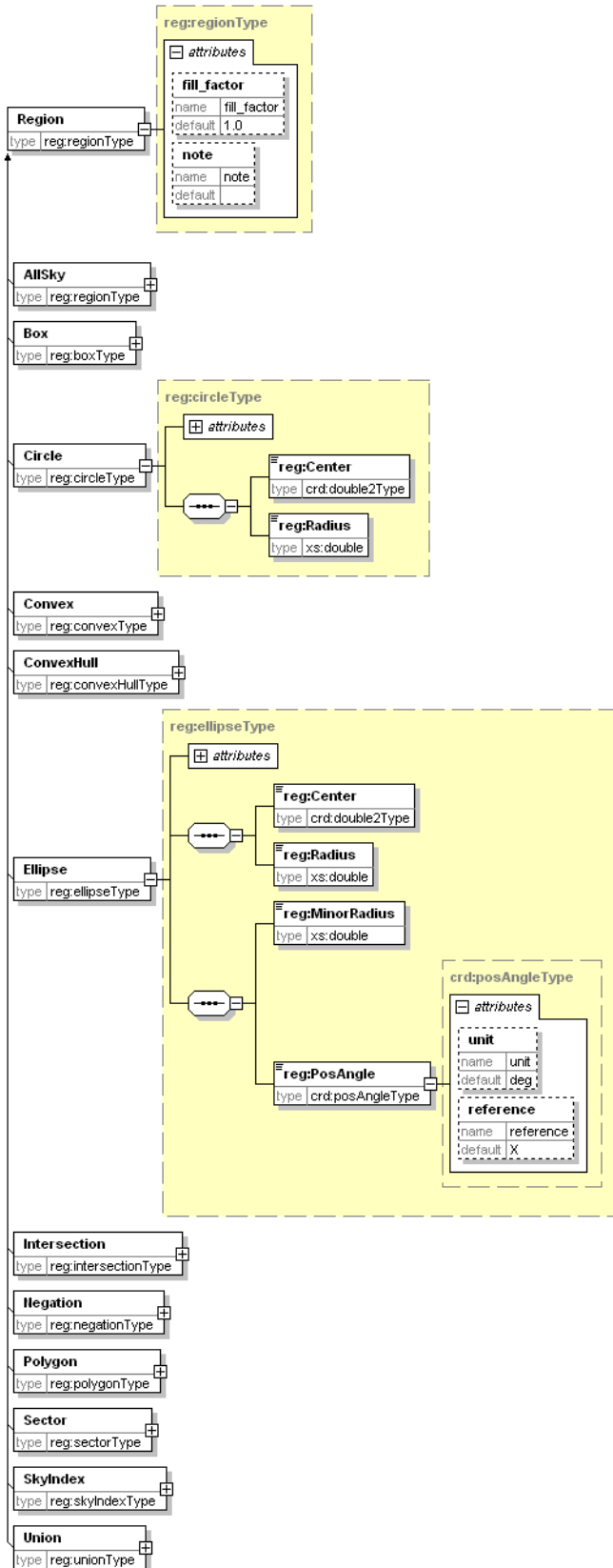
## Spatial Region

Region is a specialized construct to address the needs for expressing common spatial shapes in a comprehensive way. A Region may consist of a Shape or be the result of an operation performed on one (**Negation**) or two (**Union**, **Intersection**) Regions.

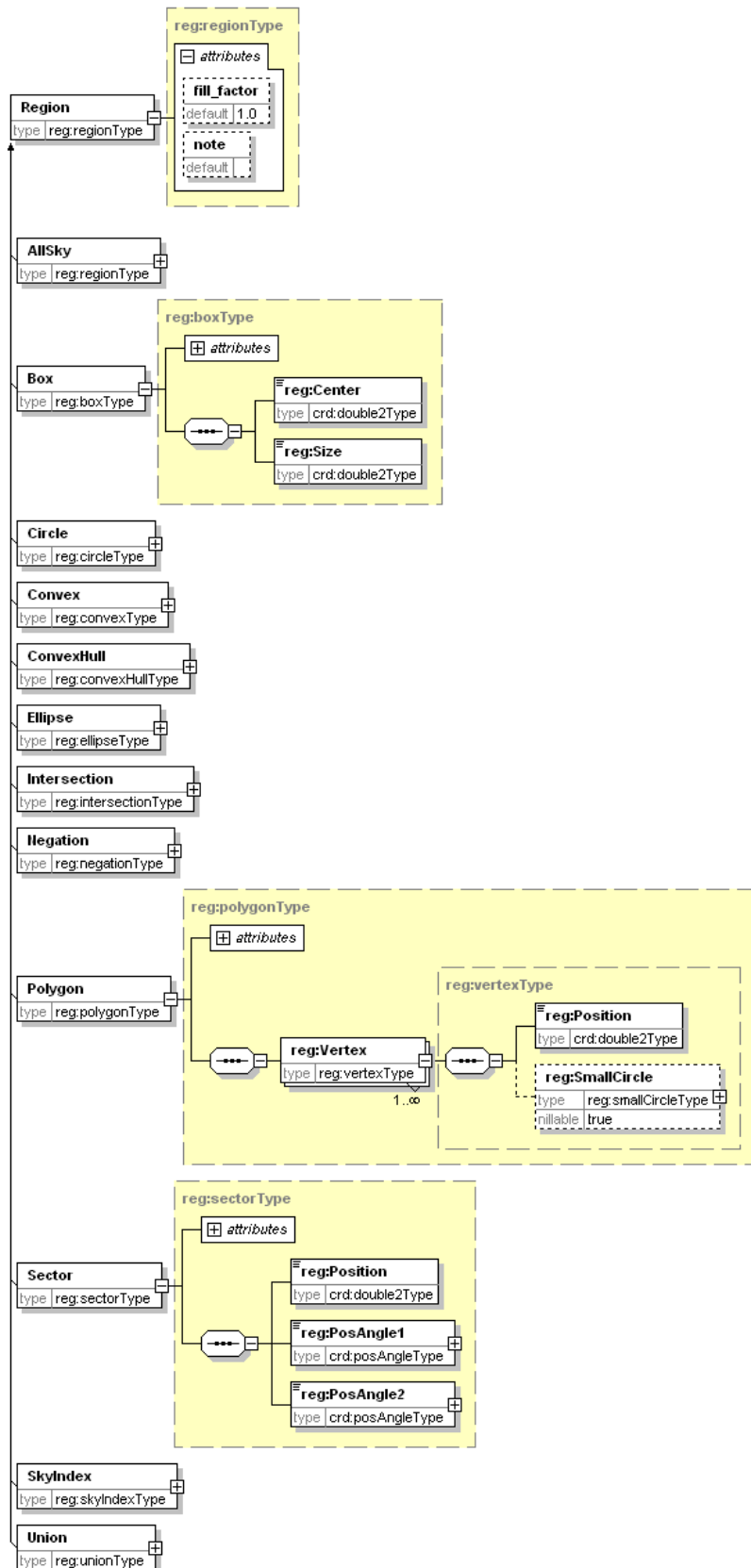


**Region: Allsky, Circle, and Ellipse**

Allsky is an empty element (except for FillFactor) that is just a convenience.



**Region: Polygon, Box, and Sector**



### Region: Convex and Convex Hull

Convex and ConvexHull operate on the Unit Sphere.

