On the creation of data models with usable utypes

Jesus Salgado
ESAVO
19/10/2011

Utypes during data model creation

- During the process of creation of a data model and protocols questions appear on the utypes creation

- Problems can be divided into three different groups:
    1. utypes definition
    2. utypes uniqueness in a document response
    3. utypes reuse between two models

- I present some of the problems I have found myself during Data Modeling and the solutions we have implemented before a clear answer on that

Utypes note rules for Data Modeling

1. Property names are unique in a Class. Note there are three types of properties: An Attribute is a property the data type of which is a value type (NOT an object type,/class), though it need not be primitive but may be structured (i.e. have attributes of its own). A Collection is a named, 1-to many composition relation of a parent to a child class. A Reference is a named, many-to-one shared association to another class
2. Class names are unique in a Package (name space)
3. Package names are unique in either an enclosing parent package, or in the Model (the root of all)

Utypes definition

What are Utypes for?

The main goal of Utypes is to help to parameterize a data model, i.e. to describe all items in the model as a list of keyword-value pairs.

Uniqueness: Utypes, UFIs, etc

- There have been many discussions on if utypes should be unique or not within a document or to allow unambiguous queries
- These questions have been answered in two ways:
  - 1. utypes are not unique within a document.
  - 2. UFIs

- Example from SLAP specification

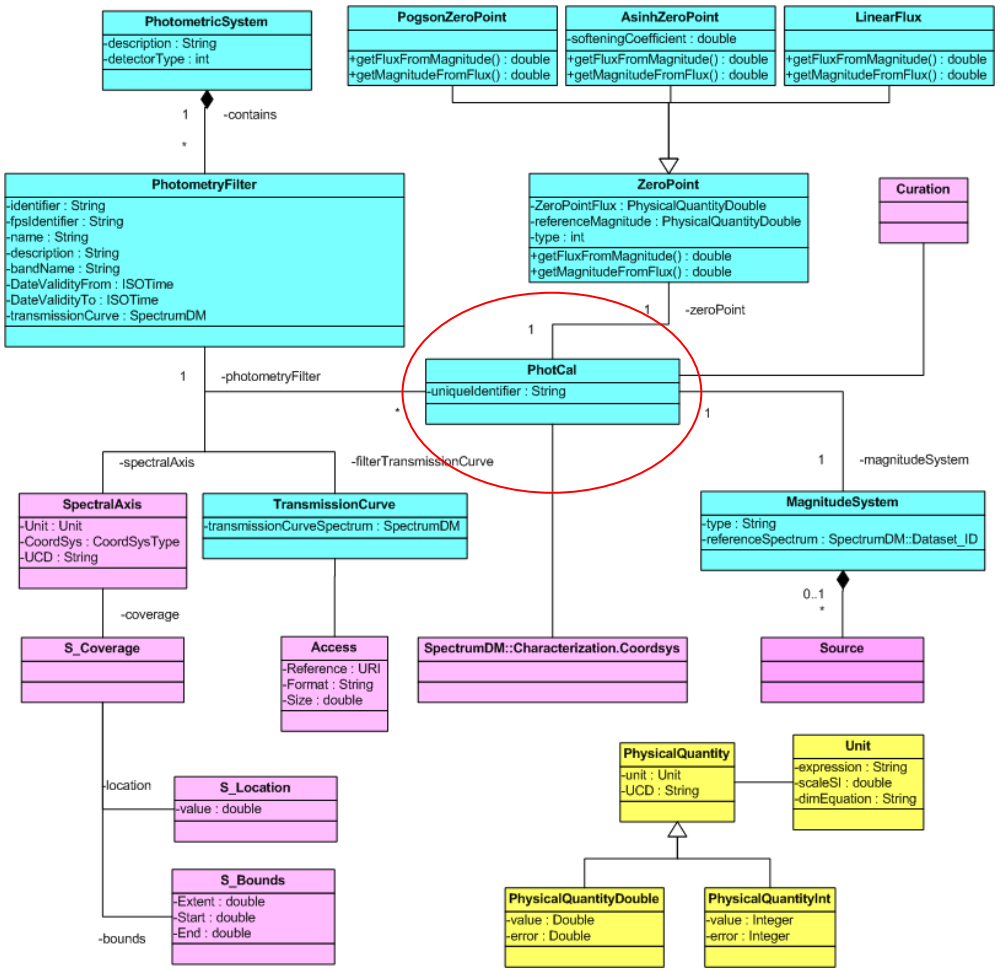
  *One field **MUST** have utype="**ssldm:Line.wavelength.value**", with datatype="double" and ucd="em.wl", containing the wavelength in vacuum of the transition originating the line in meters.*

  *It is allowed to have more than one wavelength field in different units in order to preserve the precision of the original value prior to unit conversion in the client. If this is the case, and to get backwards compatibility with already existing services, there **MUST** be one field with utype="**ssldm:Line.wavelength.value**" and unit="m" in the VOTable response.*

  Other fields with the same utype should have a different value in the unit field descriptor.
- We have used "unit" to distinguish the two fields but there are case more difficult

Uniqueness: Utypes, UFIs, etc (II)

- Data model strategy:



Uniqueness: Utypes, UFIs, etc (III)

PhotCal.identifier	meta.ref.ivorn	Unique identifier of the Photometry Calibration instance within a FPS		string
PhotCal.ZeroPoint.Flux.unit	meta.unit	unit for Zero point flux	Jy	string
PhotCal.ZeroPoint.Flux.ucd	meta.ucd	ucd for Zero point flux	phot.flux.density	string
PhotCal.ZeroPoint.Flux.value	phot.flux.density	flux value at Zero point associated to this filter		Double
PhotCal.ZeroPoint.Flux.error	phot.flux.density; stat.error	Error in the flux value at Zero point associated to this filter		Double
PhotCal.ZeroPoint.referenceMagnitude.value	phot.mag	Reference magnitude used for zero point	0.0	Double
PhotCal.ZeroPoint.referenceMagnitude.error	phot.mag;stat.error	Error in the reference magnitude used for zero point	0.0	Double
PhotCal.ZeroPoint.type	meta.code	Type of zero point	0.0	enum int
PhotCal.AsinhZeroPoint.softeningParameter	obs.param	Correction parameter for luptitudes	0	double
PhotCal.MagnitudeSystem.type	meta.code	Type of magnitude system	VEGAMag	string
PhotCal.MagnitudeSystem.ReferenceSpectrum.uri	meta.ref.ivorn	Reference SED or spectrum for this magnitude system		uri type

Uniqueness: Utypes, UFIs, etc (IV)

- Is this approach scalable? Obviously not in some cases
- STC (a complex DM) produces ambiguous utypes

SpatialAxis.Coverage.location.coord;stc:Position2D.Value2D.C1
SpatialAxis.Coverage.location.Position2D.Value2D.C1

- Obviously, this can appear for different fields in an output response (for different Coordinates Reference Frames, Epochs, etc) although this could be relaxed by the use of GROUPs/PARAMs (see Markus Demleitner et al note)
- Can be used for queries? Only if all the necessary info is set in the query or if a protocol is defined so coordinate reference frame, epoch, etc is already fixed.
- Is this against TAP? Can utypes used for queries?????

Uniqueness: Utypes, UFIs, etc (V)

- UFIs: If we need unique utypes within a document or in a query, we need to use UFIs. Syntax was a XPath like:

  UFI=token1.token2[attribute1=val1].token3.token4

- UFI creation looks cumbersome for some data models although they provide extra flexibility not present in original utypes
- Are we trying to serialize an attribute/class instance in a string?
- Concept of UFIs seems to be death. No application or specification make use of them (As far as I know)
- Do we need still to work on it?
- Are UFIs still necessary?
- Do we need to include them in the formal spec?

Utypes reuse between models

- A manual mechanism has been set to reuse utypes between data models. Two different approaches described in the current note:

- Canonical representation:

  SpatialAxis.Coverage.location.coord;stc:Position2D.Value2D.C1
- Alternative representation (although the most extended)

  SpatialAxis.Coverage.location.Position2D.Value2D.C1

- However, we would like to have a clear machine readable meaning for the utypes present in a response
- Perhaps, dynamic interpretation of utypes is not necessary

Utypes reuse between models

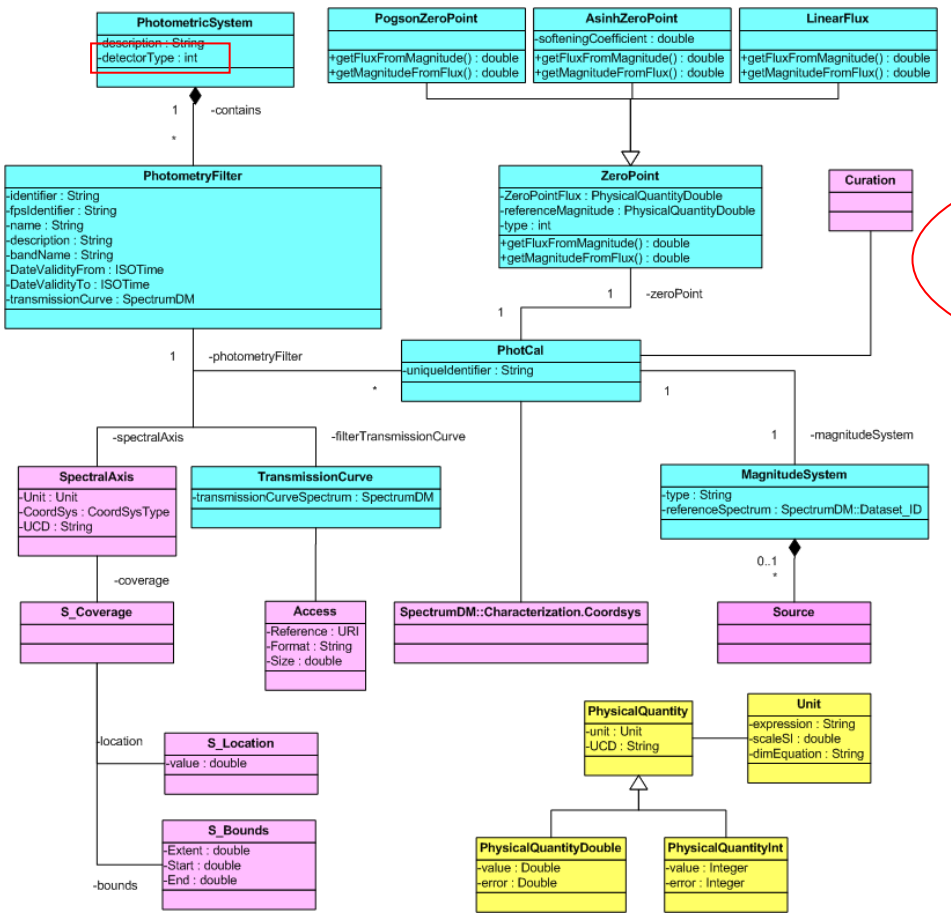
Problems found:

1. First Data Models were concentrated in specific use cases e.g. observation description, spectral description, etc. All the elements needed were created within the DMs. It is difficult to reuse elements are elements are classes within the model (One good counterexample in Characterization where many elements can be reused)

2. Navigation of objects. It could require a unique/main class and attach objects at the end of the branch (this is not always possible)

Creation of reusable elements could limit this problem

Utypes reuse between models (II)



Reusable approach:

photDM:PhotometricSystem.detectorType

Use case consistent approach:

photDM:PhotometryFilter.
PhotometricSystem.detectorType

Data Model main class approach:

photDM:PhotCal.PhotometryFilter.
PhotometricSystem.detectorType

Utypes note subjects the use of packages…

photDM:photcal/photometryfilter/
PhotometricSystem.detectorType

Conclusions

- A data model can be written in a way that utypes are extracted easily
- However, this depends a lot of the complexity of the data model

- How much unique is a utype?
- Could be use utypes in queries?
- How to reuse them? Should we recommend rules so utypes can be easily reusable?

- If a clear utype definition and use cases to be fulfilled is clearly establish, the definition of a set of rules in the utypes creation could be needed

THANK YOU

Jesus Salgado

Utypes: On the creation of DM with reusable utypes

Jesus.Salgado@sciops.esa.int