VO-DML Mapping

Concepts

Graphical mapper tool

Syntax summary

(VO-DML) mapping

- Expresses how instances of a data model (expressed as VO-DML) are represented in a tabular representation.
  - VOTable
  - TAP_SCHEMA
- So assumes that data models used in a mapping are defined explicitly in VO-DML/XML documents that can be referenced

http://ivoa.net/documents/VODML/20180910

IVOA Recommendation

International Virtual Observatory Alliance

IVOA Documents

VO-DML: a consistent modeling language for IVOA data models
Version 1.0

IVOA Recommendation 10 September 2018

Interest/Working Group:
http://www.ivoa.net/twiki/bin/view/IVOA/IvoaDataModel
Author(s):
Gerard Lemson, Omar Laurino, Laurent Bourges, Mark Cresitello-Dittmar, Markus Demleitner, Tom Donaldson, Patrick Dowler, Matthew Graham, Norman Gray, Laurent Michel, Jesus Salgado
Editor(s):
Gerard Lemson, Omar Laurino
DOI:
10.5479/ADS/bib/2018ivoa.spec.0910L

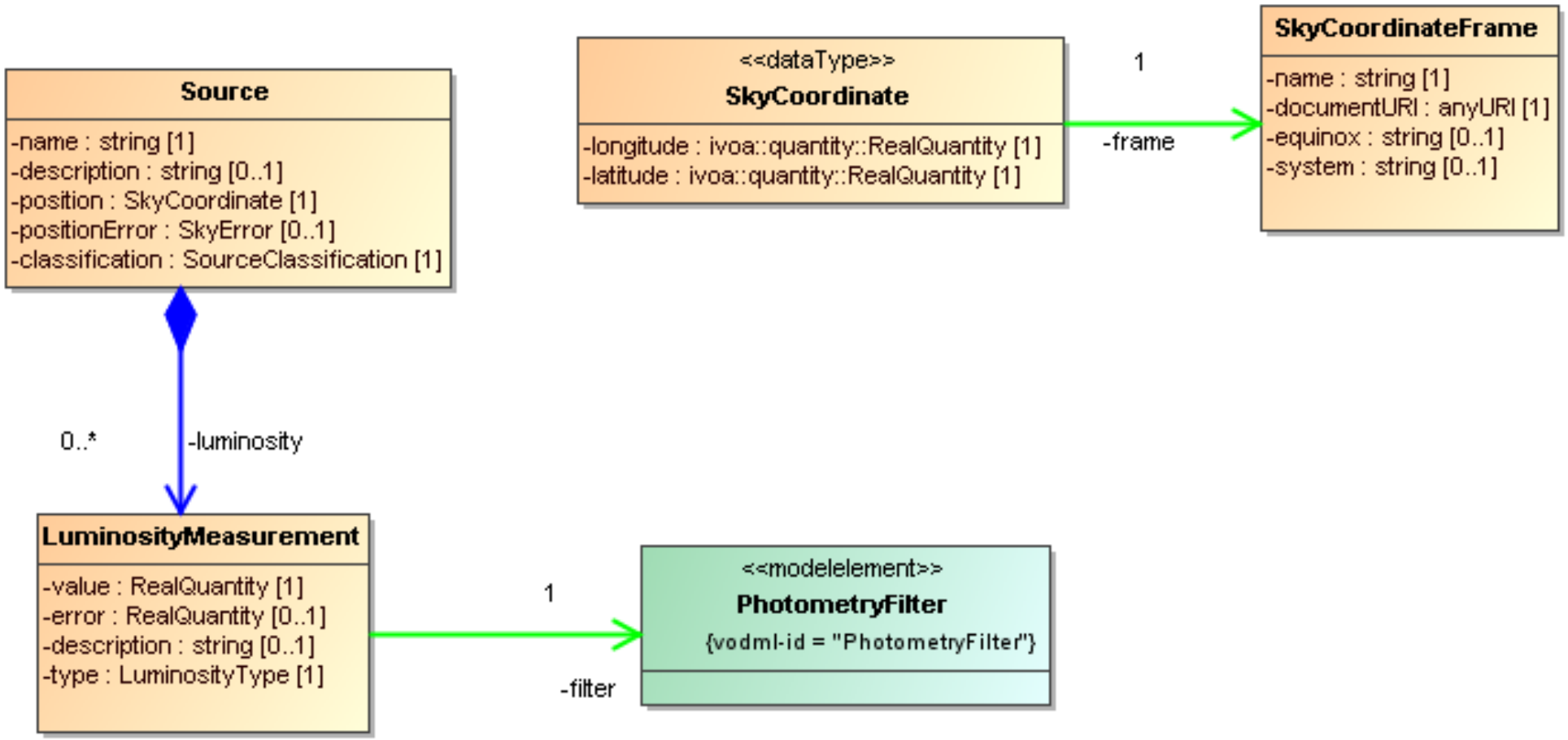
Supporting Mapping was motivation and one of the core goals of VO-DML:

From section 2, p12:

VO-DML is designed to satisfy the following requirements. It should

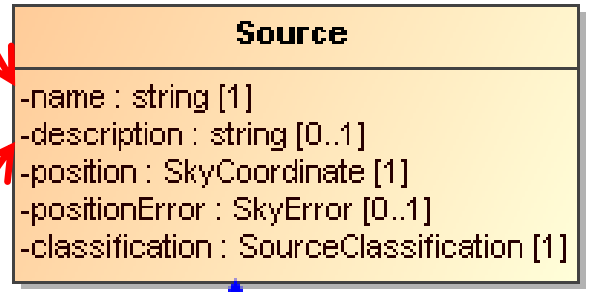
1. *Support the specification of serialization strategies for serializing instances of data models into different file formats;*
2. *Be rich enough to represent existing IVOA data models;*
3. *Support model reuse;*
4. *Be implementation-neutral, but...*
5. *Be flexible enough to be mapped to important physical representations, in particular XML schema, relational model (TAP), object-oriented languages (Java, Python...), and at the same time...*
6. *Be as minimal as possible, avoiding redundancy, adding restrictions where possible, with the aim of simplifying the work of modelers by offering few and "obvious" choices;*
7. *Be based on accepted standards for data modeling, but ...*
8. *Not rely on external modeling tools, but be sufficiently compatible with them so that such tools MAY be used when representing models;*
9. *Support runtime model interpretation;*

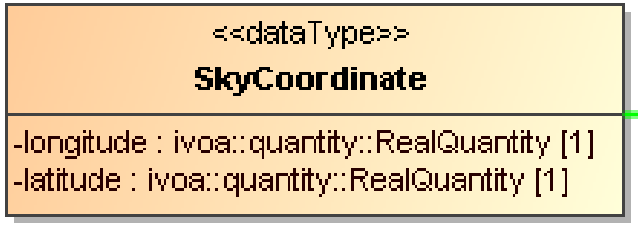
Example: Simple source data model

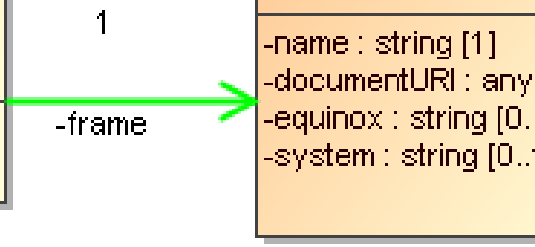


objid	ra	dec	u	g	r	i	z	run	rerun	camcol	field	specobjid	class	redshift	plate	mjd	fiberid
1237680191504712292	319.42017295	-2.91605515	19.453272	17.512213	16.45					5	45	4933710530549838848	GALAXY	0.091911	4382	55742	62
1237680191504842797	319.6726666	-2.89320328	18.006258	18.408295	17.02					5	47	4933700634945188864	STAR	-9.116632E-5	4382	55742	26
1237660241388240997	51.95792979	0.44178806	17.90674	16.767498	16.24					5	146	2329593891403098112	STAR	-2.435169E-4	2069	53376	389
1237660241388371981	52.15864799	0.5100779	18.619341	17.314531	16.76					5	148	2329602687496120320	STAR	-1.707261E-5	2069	53376	421
1237660241925505040	52.72667419	0.88746662	17.930399	16.900446	16.418163	16.214106	16.1061	3438	301	6	152	2329600488472864768	STAR	-1.622409E-4	2069	53376	413
1237660241925505156	52.85661769	0.97756273	18.178764	16.997499	16.512629	16.314194	16.207306	3438	301	6	152	2329599663839143936	STAR	-1.234436E-4	2069	53376	410
1237662303111507089	202.55299093	39.86892911	17.820675	16.164869	15.296254	14.812856	14.419583	3919	301	1	16	5299625250001449984	GALAXY	0.048569	4707	55653	52
1237663238729787997	52.05059022	0.14966321	19.351822	18.277271	18.06134	17.999191	17.999123	4136	301	4	165	2329595265792632832	STAR	-5.184785E-4	2069	53376	394
1237651271358408122	158.78373508	63.9613952	19.283352	17.41073	16.419657	16.042131	15.731997	1350	301	1	295	550602195343534080	GALAXY	0.11802	489	51930	135
1237651271358801125	158.82992158	63.94061555	19.297565	17.409573	16.431635	16.044048	15.707916	1350	301	1	295	550601370709813248	GALAXY	0.117888	489	51930	132

SDSS





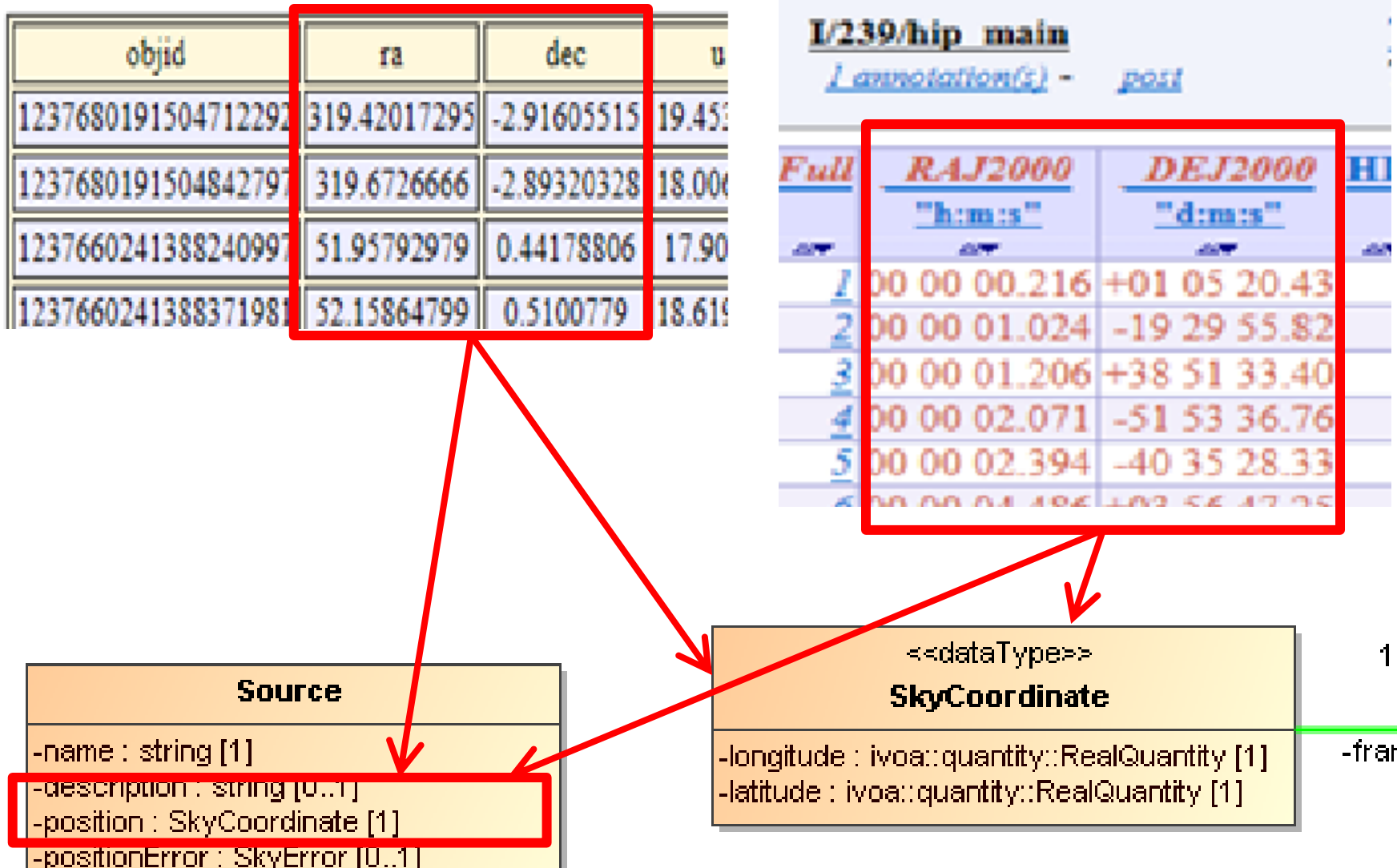


Identifying that a table contains **Sources**

I/239/hip_main — The Hipparcos and Tycho Catalogues (ESA 1997)
1 annotation(s) - *post* — The Hipparcos Main Catalogue *(118218 rows)*

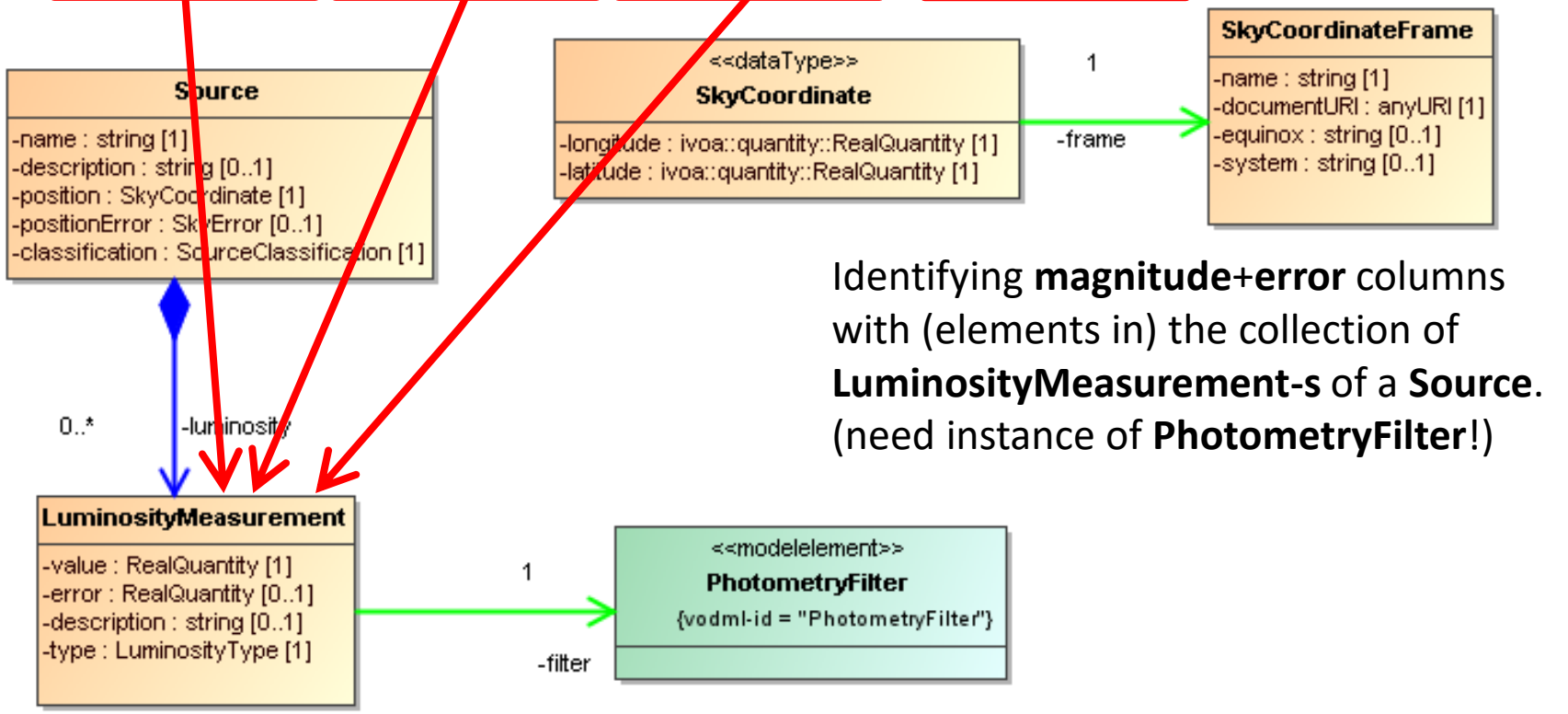
Full	RAJ2000 "h:m:s"	DEJ2000 "d:m:s"	HIP	RAhms	DEdms	Vmag mag	RA(ICRS) deg	DE(ICRS) deg	Plx mas	pmRA mas/yr	pmDE mas/yr	e_Plx mas	BTmag mag	e_ mag	VTmag mag	e_ mag	B-V mag	Hpmag mag	e_ mag
1	00 00 00.216	+01 05 20.43	1	00 00 00.22	+01 05 20.4	9.10	0.00091185	1.08901332	3.54	-5.20	-1.88	1.39	9.643	0.020	9.130	0.019	0.482	9.2043	0.0020
2	00 00 01.024	-19 29 55.82	2	00 00 00.91	-19 29 55.8	9.27	0.00379737	-19.49883745	21.90	181.21	-0.93	3.10	10.519	0.033	9.378	0.021	0.999	9.4017	0.0017
3	00 00 01.206	+38 51 33.40	3	00 00 01.20	+38 51 33.4	6.61	0.00500795	38.85928608	2.81	5.24	-2.91	0.63	6.576	0.004	6.621	0.005	-0.019	6.6081	0.0007
4	00 00 02.071	-51 53 36.76	4	00 00 02.01	-51 53 36.8	8.06	0.00838170	-51.89354612	7.75	62.85	0.16	0.97	8.471	0.007	8.092	0.007	0.370	8.1498	0.0011
5	00 00 02.394	-40 35 28.33	5	00 00 02.39	-40 35 28.4	8.55	0.00996534	-40.59122440	2.87	2.53	9.07	1.11	9.693	0.014	8.656	0.010	0.902	8.7077	0.0018
6	00 00 04.486	+03 56 47.25	6	00													1.336	12.4488	0.0085
7	00 00 05.283	+20 02 10.01	7	00									.542	0.039	9.679	0.030	0.740	9.6795	0.0021
8	00 00 06.562	+25 53 11.26	8	00									.433	0.055	9.151	0.029	1.102	8.5522	0.1671
9	00 00 08.477	+36 35 09.45	9	00									.962	0.025	8.711	0.015	1.067	8.7534	0.0018
10	00 00 08.740	-50 52 01.11	10	00									.140	0.011	8.630	0.010	0.489	8.6994	0.0020
11	00 00 08.961	+46 56 23.99	11	00 00 08.95	+46 56 24.0	7.34	0.03729695	46.94000154	4.29	11.09	-2.02	0.84	7.446	0.005	7.364	0.005	0.081	7.3777	0.0010
12	00 00 09.816	-35 57 36.81	12	00 00 09.82	-35 57 36.8	8.43	0.04091756	-35.96022482	4.06	-5.99	-0.10	1.16	10.369	0.023	8.588	0.010	1.484	8.5598	0.0012
13	00 00 10.008	-22 35 40.94	13	00 00 10.00	-22 35 40.9	8.80	0.04167970	-22.59468060	3.49	8.45	-10.07	1.48	10.216	0.026	8.887	0.014	1.128	8.9707	0.0017

Hipparcos@VizieR



Identifying two columns as the **position** of a **Source**, a **SkyCoordinate**

ec	u	g	r	i	z	ru
505515	19.453272	17.512213	16.457823	15.911284	15.466995	808
320328	18.006258	18.408295	17.027491	16.594183	15.47355	808
78806	17.90674	16.767498	16.243202	16.038309	15.933592	343
00779	18.619341	17.314531	16.763399	16.53455	16.394312	343
46662	17.930399	16.900446	16.418163	16.214106	16.1061	343



Identifying **magnitude+error** columns with (elements in) the collection of **LuminosityMeasurement-s** of a **Source**. (need instance of **PhotometryFilter**!)

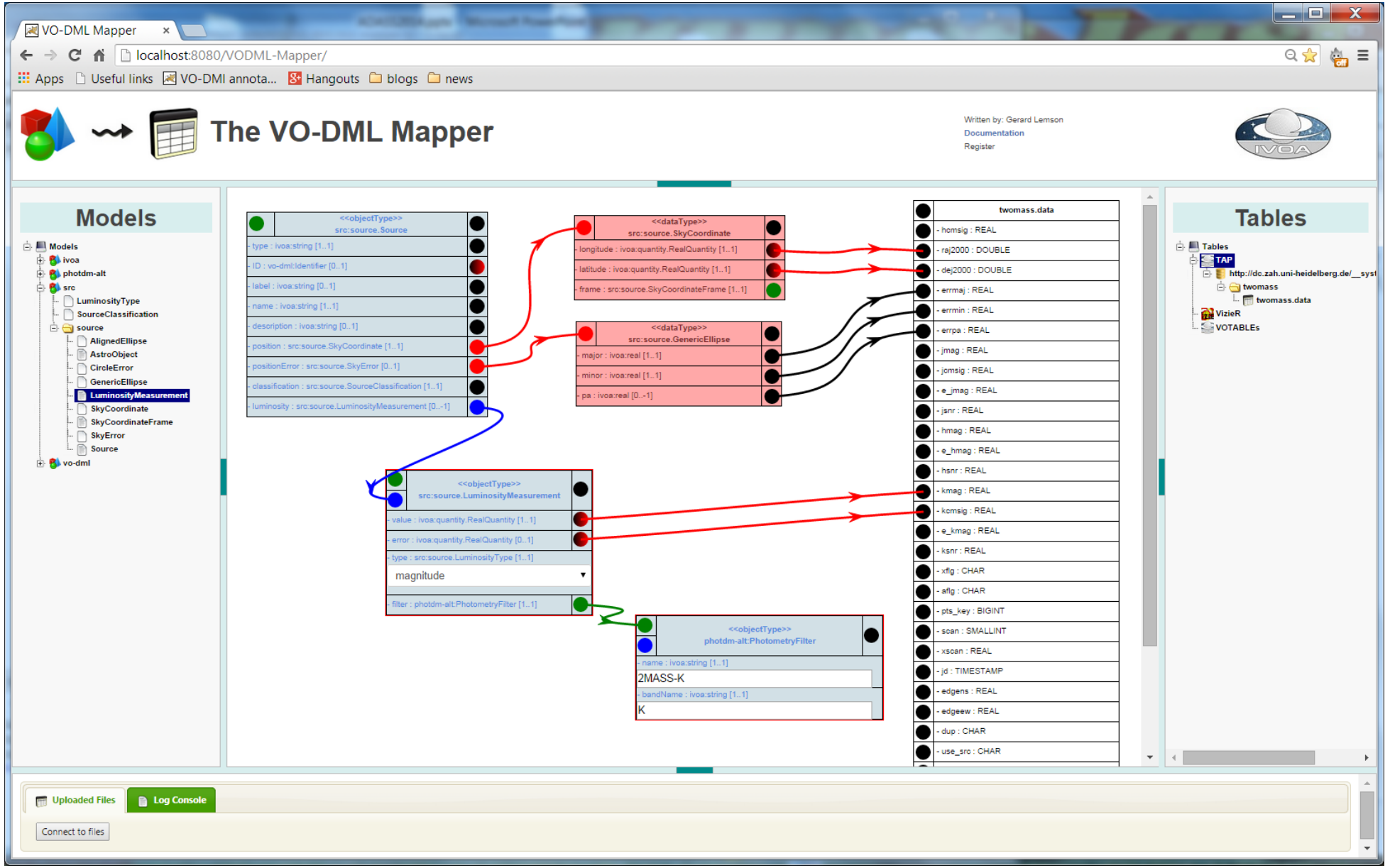
How to express this?

- Many words spent on syntax
- We went through quite a few iterations to accommodate requests
  - Starting with UTYPEs tiger team
  - Later mainly Omar, Mark CD, Tom, Laurent and me
- Even without any additions to VOTable schema mapping was deemed complex
- Early comment (M. Juric) asked about a tool that could help

VO-DML Mapper

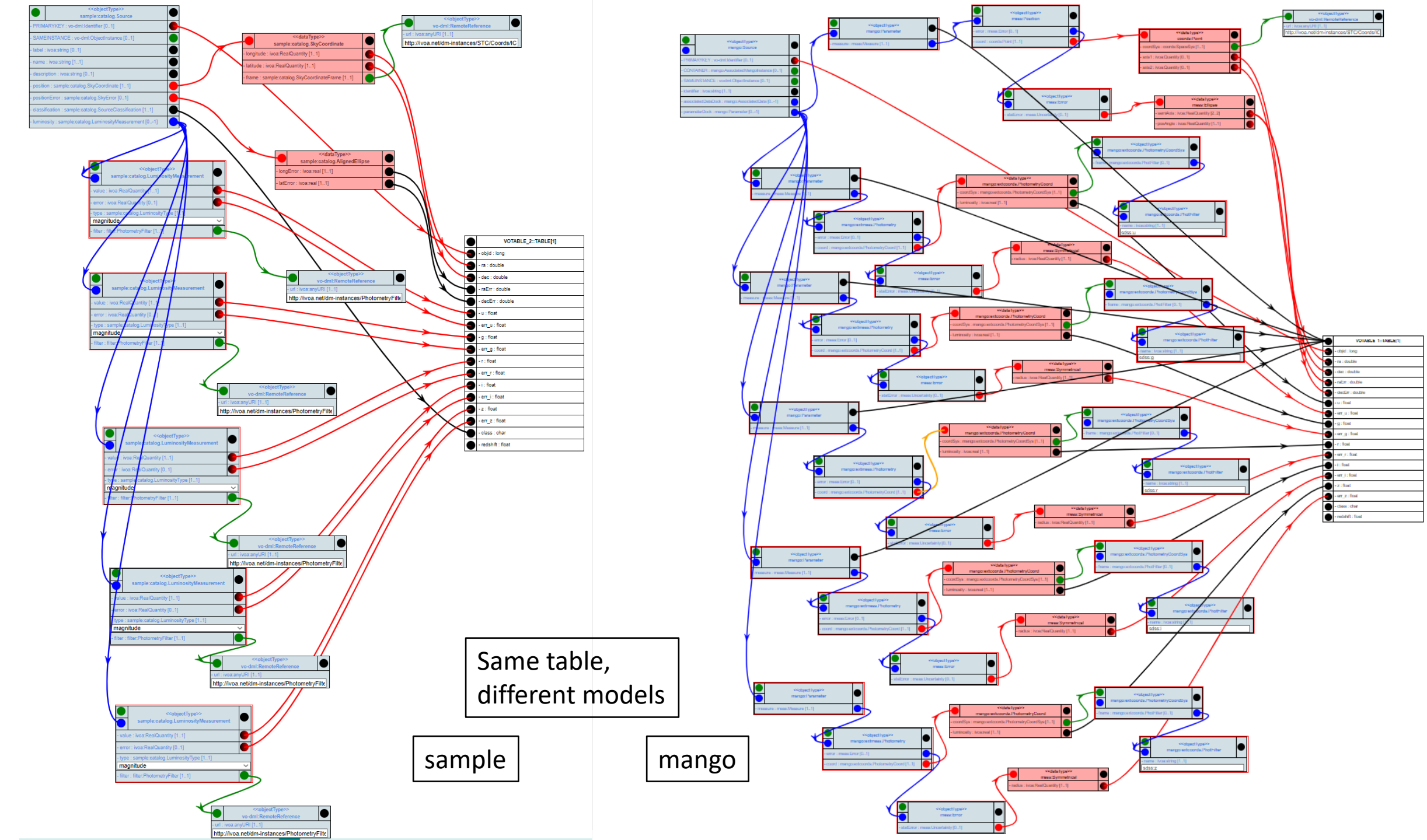
- A tool that
  - can read TAP_SCHEMAS, VOTables
  - can read VO-DML data models
  - can draw tables and “serialization instances” on a canvas
  - allows elements to be linked, mapped
  - can generate VO-DML Mapping according to the original syntax
- Was used in dm-usecases
- Quick demo

http://dsa012.pha.jhu.edu:8081/VODML-Mapper



It is not the syntax that makes mapping complex!

It’s the impedance mismatch between model and tabular representation.


Same table,
different models
sample
mango

On to Syntax

The spec and the syntax

- Spec (ivoatex under construction): https://github.com/ivoa/mapping-vodml/tree/port-to-ivoatex
- XSD:
  inside VOTable xsd:
  http://volute.g-vo.org/svn/trunk/projects/dm/vo-dml/xsd/votable_ext/VOTable-1.4_vodml.xsd
  external, to be imported by VOTable:
  http://volute.g-vo.org/svn/trunk/projects/dm/vo-dml/xsd/votable_ext/VODML-mapping.xsd more comparable to LMs xsd
- Btw, long history of documents, schemas, sample documents and tools starting 2012/13

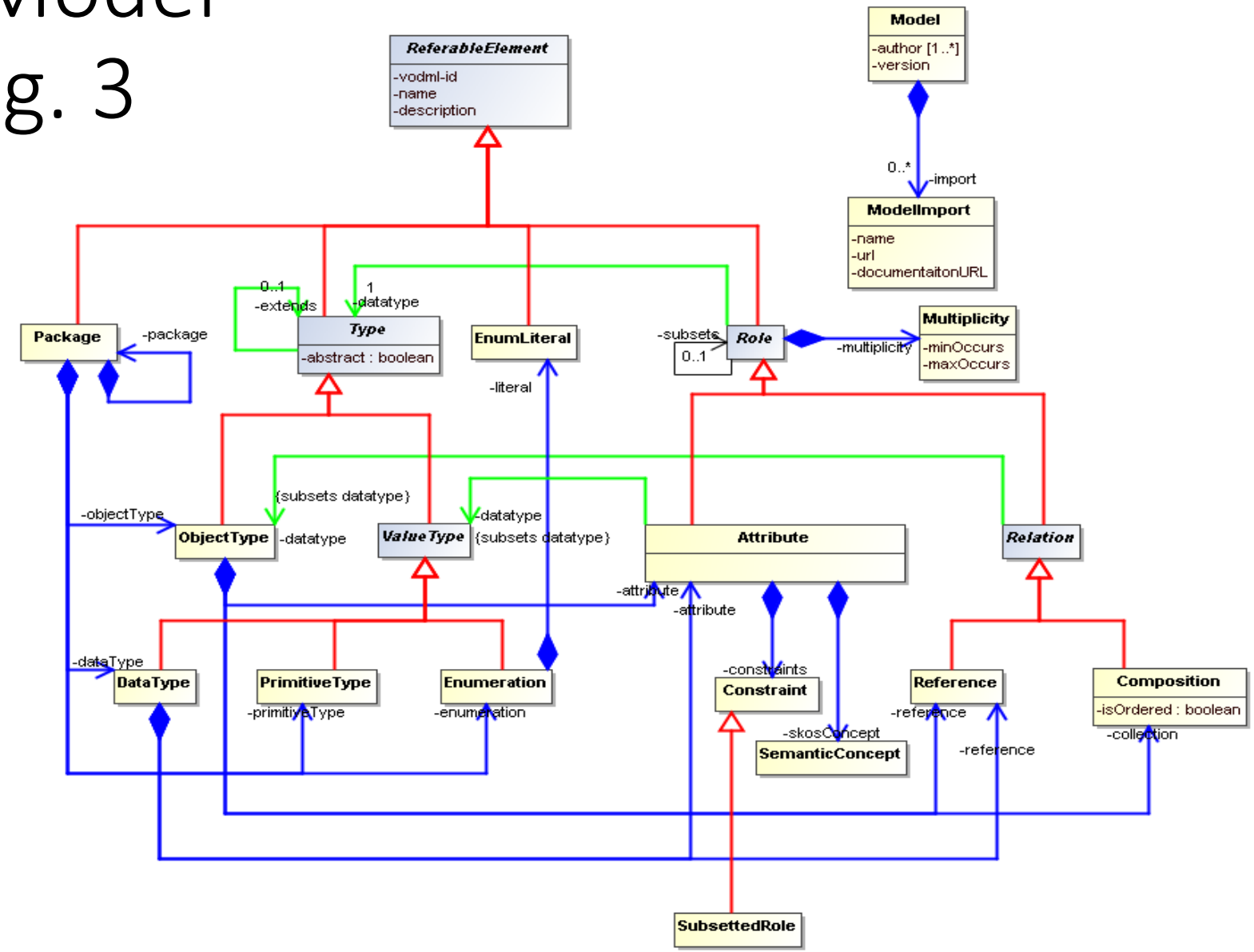
Mapping Meta-Models

- VO-DML is IVOA standard meta-model/language for structure of data models
- VOTable is IVOA standard meta-model/language for structure of files containing tabular data sets
- TAP_SCHEMA is IVOA standard-meta model/language for structure of relational databases

Goals

1. Express mapping at the meta-model level
   - As much as possible link concepts from one meta-model to another
2. Make the mapping as much as possible 1-1
   - Should facilitate interpretation
3. Be inspired by existing practice
   - Object relational mapping standards ala JPA

VO-DML Meta-Model
VO-DML spec Fig. 3



Inspired by object-relational mapping VO-DML spec Appendix B

VO-DML concept	RDB concept
ObjectType	`Table`
DataType	Used as data type of Attributes, mapped to one or more `Columns` in a `Table`.
Enumeration	Used as data type of Attributes, mapped to one `Column` in a `Table`
PrimitiveType	Used as data type of Attributes, mapped to one `Column` in a `Table`
Attribute	One or more `Columns` in a `Table` depending on datatype
Composition	`Foreign key` from child to parent `Table`
Reference	`Foreign key` from referrer to referent.
Type.extends	Depending on inheritance mapping strategy[41] this can be a `foreign key` from a `Table` representing the sub-type to the one representing the base type. Or may lead to sub-type `Columns` being added to table for super-type.
Package	Could be mapped to a `Schema`, but should have no nested packages as nesting of schemas is not supported in RDBs.

VO-DML Mapping

https://volute.g-vo.org/svn/trunk/projects/dm/vo-dml/xsd/votable_ext/VODML-mapping.xsd

```
<VODML>
⊕ <MODEL>
⊕ <GLOBALS>
⊕ <TEMPLATES>
</VODML>
```

- VO-DML
- Serialization
- VOTable

```
<COMPOSITION
  @dmrole >
 ⊕ <INSTANCE>
 ⊕ <EXTINSTANCES>
</COMPOSITION>
```

```
<COLUMN
 @dmtype
 @ref >
⊕ <OPTIONMAPPING>
</COLUMN>
```

```
<MODEL>
⊕ <NAME>
⊕ <URL>
⊕ <IDENTIFIER>
</MODEL>
```

```
<ATTRIBUTE
  @dmrole >
(⊕(<COLUMN>|<CONSTANT>|<LITERAL>) |
 ⊕ <INSTANCE> )
</ATTRIBUTE>
```

```
<CONSTANT
 @dmtype
 @ref >
⊕ <OPTIONMAPPING>
</CONSTANT>
```

```
<REFERENCE
  @dmrole >
 ⊕(<IDREF>|<REMOTEREFERENCE>|<FOREIGNKEY>)
</REFERENCE>
```

```
<LITERAL
 @dmtype
 @value
 @unit >
⊕ <OPTIONMAPPING>
</LITERAL>
```

```
<GLOBALS
  @ID >
⊕ <INSTANCE>
</GLOBALS>
```

```
<INSTANCE
  @dmtype
  @ID >
⊕ <PRIMARYKEY>
⊕ <CONTAINER>
⊕ <ATTRIBUTE>
⊕ <COMPOSITION>
⊕ <REFERENCE>
</INSTANCE>
```

```
<CONTAINER
  @dmrole >
 (<IDREF>|<REMOTEREFERENCE>|<FOREIGNKEY>)
</CONTAINER>
```

```
<OPTIONMAPPING>
 <MAPPEDOPTION>
 ( ENUMLITERAL | SEMANTICCONCEPT>)
</OPTIONMAPPING>
```

```
<TEMPLATES
 @tableref >
⊕ <INSTANCE>
</TEMPLATES>
```

```
<PRIMARYKEY>
⊕ <PKFIELD>
</PRIMARYKEY>
```

```
<FOREIGNKEY>
⊕ <PKFIELD>
  <TARGETID>
</FOREIGNKEY>
```

```
@dmrole/@dmtype/<ENUMLITERAL>: xsd:string
[a-zA-Z][a-zA-Z0-9_\-]*:[a-zA-Z][a-zA-Z0-9\._]*
```

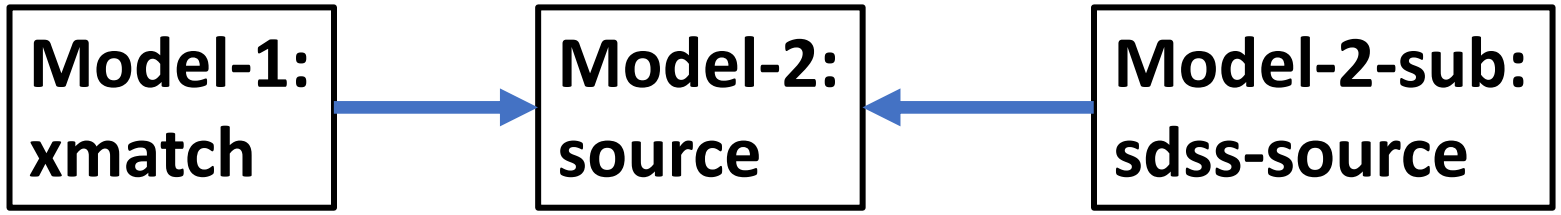
```
<PKFIELD>
 (<COLUMN>|<CONSTANT>|<LITERAL>)
</PKFIELD>
```

VODML/MODEL

```
<VODML>
⊕ <MODEL>
⊕ <GLOBALS>
⊕ <TEMPLATES>
</VODML>
```

```
<MODEL>
⊕ <NAME>
⊕ <URL>
⊕ <IDENTIFIER>
</MODEL>
```

- Single VODML element directly under VOTablev contains all annotations
- MODEL/URL identifies VO-DML/XML document.
- MODEL/NAME MUST be the VO-DML Model.name and must be used as prefix for @vodml-ref references
- There MUST be one MODEL declaration for each VO-DML Model that is used in the VOTable.
  - Follows XML prefix declaration approach
  - Also not guaranteed that all models can be obtained from a single “root” model…



- And people familiar with Model-2(-sub) for example need not also understand/dynamically parse model 1 to find out whether they may expect instances form their model

VOTABLE/VODML/MODEL

```
<VOTABLE
    xmlns="http://www.ivoa.net/xml/VOTable/v1.4_vodml"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.ivoa.net/xml/VOTable/v1.4_vodml https://volute.g-vo.org/svn/trunk/projects/dm/vo-dml-c
    <VODML>
        <MODEL>
            <NAME>mango</NAME>
            <URL>https://raw.githubusercontent.com/ivoa-std/MANGO/a46441f6fc498a6aeb33ed97e65689fee3d00f6c/vo-dml/mango.v
        </MODEL>
        <MODEL>
            <NAME>meas</NAME>
            <URL>http://volute.g-vo.org/svn/trunk/projects/dm/STC/Meas/vo-dml/STC_meas-v1.0.vo-dml.xml</URL>
        </MODEL>
        <MODEL>
            <NAME>coords</NAME>
            <URL>http://volute.g-vo.org/svn/trunk/projects/dm/STC/Coords/vo-dml/STC_coords-v1.0.vo-dml.xml</URL>
        </MODEL>
        <MODEL>
            <NAME>ivoa</NAME>
            <URL>http://volute.g-vo.org/svn/trunk/projects/dm/vo-dml/models/ivoa/vo-dml/IVOA-v1.0.vo-dml.xml</URL>
        </MODEL>
```

GLOBALS

```
<VODML>
⊕ <MODEL>
⊕ <GLOBALS>
⊕ <TEMPLATES>
</VODML>
```

- container for “standalone” instances of VO-DML types
- Generally reference data used later on in the document
- Ex: defining coordinate systems, filter definitions
- Ex: defining a “singleton” Cube object, contents of which in a TABLE below

- NB: at some point we considered having one GLOBALS container per table.
  - Seemed unnecessary complication.
  - code does not care where a referenced INSTANCE resides.
  - how would one decide what object goes where *in an automated context*, e.g. when annotating TAP query result?

```
<GLOBALS
  @ID >
⊕ <INSTANCE>
</GLOBALS>
```

VODML/GLOBALS

```
<VODML>
...
    <GLOBALS>
            <INSTANCE dmtype="coords:SpaceSys" ID="T1616959090495_54">
                <COMPOSITION dmrole="coords:PhysicalCoordSys.coordSpace">
                    <INSTANCE dmtype="coords:SphericalCoordSpace" ID="T1616959090495_71">
                        <COMPOSITION dmrole="coords:CoordSpace.axis">
                            <INSTANCE dmtype="coords:ContinuousAxis" ID="T1616959090495_86">
                                <ATTRIBUTE dmrole="coords:Axis.name">
                                    <LITERAL dmtype="ivoa:string" value="ra"/>
                                </ATTRIBUTE>
                            </INSTANCE>
                            <INSTANCE dmtype="coords:ContinuousAxis" ID="T1616959090495_107">
                                <ATTRIBUTE dmrole="coords:Axis.name">
                                    <LITERAL dmtype="ivoa:string" value="dec"/>
                                </ATTRIBUTE>
                            </INSTANCE>
                            <INSTANCE dmtype="coords:ContinuousAxis" ID="T1616959090495_128">
                                <ATTRIBUTE dmrole="coords:Axis.name">
                                    <LITERAL dmtype="ivoa:string" value="r"/>
                                </ATTRIBUTE>
                            </INSTANCE>
                        </COMPOSITION>
                    </INSTANCE>
```

TEMPLATES

```
<VODML>
⊕ <MODEL>
⊕ <GLOBALS>
⊕ <TEMPLATES>
</VODML>
```

- Container for INSTANCES stored in rows TR
  - Every instance defined directly under template is assumed to be serialized in each row.
- One per table specified by @tableref

NB:

- INSTANCE-s need not be completely represented in row
- Structured ATTRIBUES and child INSTANCEs from composition can be stored in same row as parent
- Same INSTANCE can be present in multiple rows. Identified by PRIMARYKEY definitions

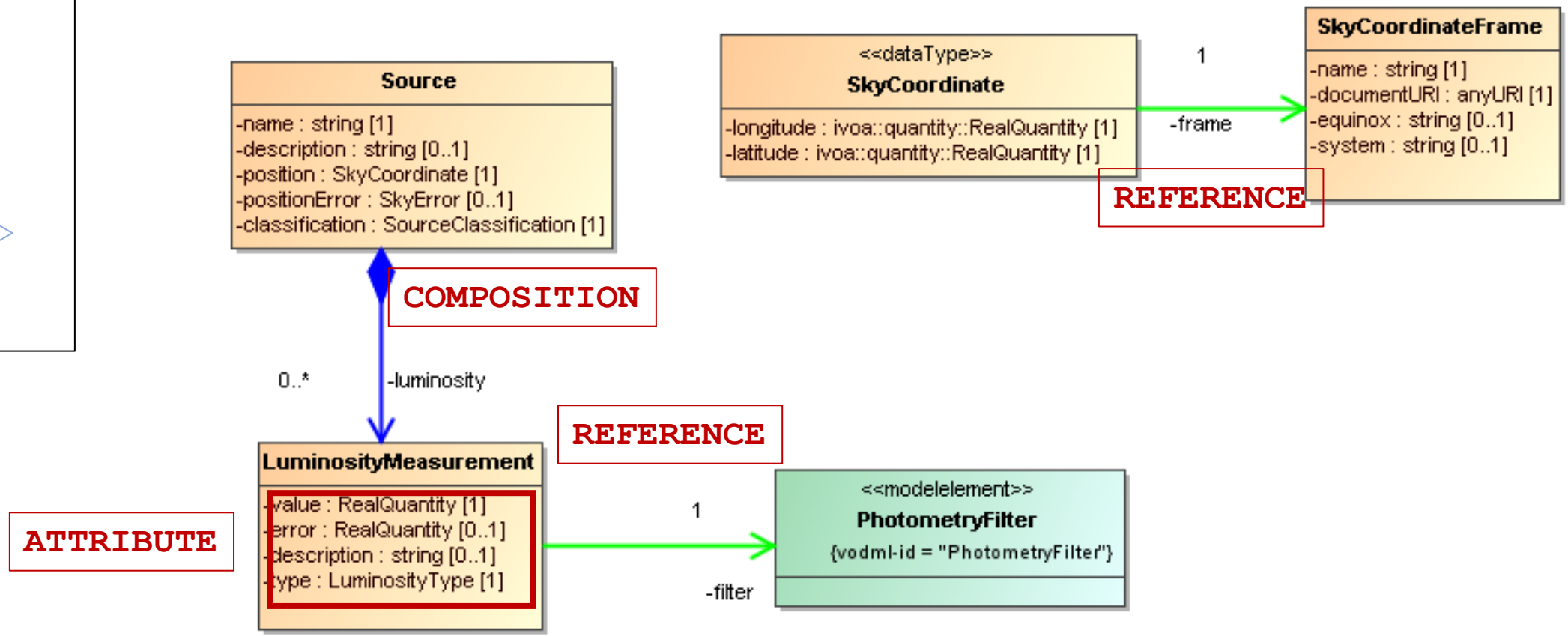
```
<TEMPLATES
 @tableref >
⊕ <INSTANCE>
</TEMPLATES>
```

VODML/TEMPLATES

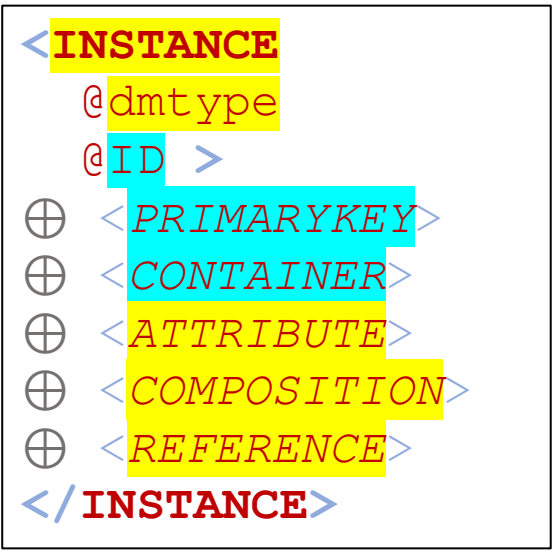
```
<VODML>
...
    <TEMPLATES tableref="T1616959090495_1">
            <INSTANCE dmtype="mango:Source" ID="T1616959090495_177">
                <PRIMARYKEY>
                    <PKFIELD>
                        <COLUMN dmtype="ivoa:string" ref="T1616959090495_1_F0"/>
                    </PKFIELD>
                </PRIMARYKEY>
                <ATTRIBUTE dmrole="mango:Source.identifier">
                    <COLUMN dmtype="ivoa:string" ref="T1616959090495_1_F0"/>
                </ATTRIBUTE>
                <COMPOSITION dmrole="mango:Source.parameterDock">
                    <INSTANCE dmtype="mango:Parameter" ID="T1616959090495_196">
                        <ATTRIBUTE dmrole="mango:Parameter.ucd">
                            <LITERAL dmtype="ivoa:string" value="pos.eq;meta.main;"/>
                        </ATTRIBUTE>
                        <ATTRIBUTE dmrole="mango:Parameter.description">
                            <LITERAL dmtype="ivoa:string" value="position"/>
                        </ATTRIBUTE>
                        <ATTRIBUTE dmrole="mango:Parameter.semantic">
                            <INSTANCE dmtype="mango:VocabularyTerm">
                                <ATTRIBUTE dmrole="mango:VocabularyTerm.label">
                                    <LITERAL dmtype="ivoa:string" value="#position"/>
                                </ATTRIBUTE>
                            </INSTANCE>
                        </ATTRIBUTE>
                        <COMPOSITION dmrole="mango:Parameter.measure">
                            <INSTANCE dmtype="meas:Position" ID="T1616959090495_228">
```

INSTANCE

```
<INSTANCE
   @dmtype
   @ID >
⊕  <PRIMARYKEY>
⊕  <CONTAINER>
⊕  <ATTRIBUTE>
⊕  <COMPOSITION>
⊕  <REFERENCE>
</INSTANCE>
```



INSTANCE



- Represents *serialized instance* of a VO-DML structured type, an ObjectType or DataType
- In *model* a structured type only contains Roles
- In serialization may also need
  - identifier: PRIMARYKEY
  - If child in composition relation, pointer to parent: CONTAINER
- A Role is the “role” played by another type in the definition of a structured type
- A Role is serialized to 0 or 1 container of type instances (note, *not necessarily* INSTANCEs)
  - So if cardinality >1, serialized Role may have >1 components inside

GLOBALS/INSTANCE

```
<INSTANCE dmtype="coords:SpaceFrame" ID="T1616959090495_149">
    <ATTRIBUTE dmrole="coords:SpaceFrame.spaceRefFrame">
        <LITERAL dmtype="ivoa:string" value="ICRS"/>
    </ATTRIBUTE>
    <ATTRIBUTE dmrole="coords:SpaceFrame.equinox">
        <LITERAL dmtype="coords:Epoch" value="NoSet"/>
    </ATTRIBUTE>
    <ATTRIBUTE dmrole="coords:SpaceFrame.refPosition">
        <INSTANCE dmtype="coords:StdRefLocation">
            <ATTRIBUTE dmrole="coords:StdRefLocation.position">
                <LITERAL dmtype="ivoa:string" value="NoSet"/>
            </ATTRIBUTE>
        </INSTANCE>
    </ATTRIBUTE>
</INSTANCE>
```

ATTRIBUTE

```
<ATTRIBUTE
  @dmrole >
(⊕(<COLUMN>|<CONSTANT>|<LITERAL>) |
 ⊕ <INSTANCE> )
</ATTRIBUTE>
```

```
<COLUMN
 @dmtype
 @ref >
⊕ <OPTIONMAPPING>
</COLUMN>
```

FIELDref

```
<CONSTANT
 @dmtype
 @ref >
⊕ <OPTIONMAPPING>
</CONSTANT>
```

PARAMref

```
<LITERAL
 @dmtype
 @value
 @unit >
⊕ <OPTIONMAPPING>
</LITERAL>
```

PARAM

- Represents a VO-DML ValueType on an ObjectType or DataType
- If PrimitiveType or Enumeration, can be represented by COLUMN, LITERAL or CONSTANT
  - Corr to FIELDref, PARAM or PARAMref in VOTable but trying to avoid name collision.
- Otherwise must be INSTANCE
  - Corr to GROUP
- COLUMN can only be used on INSTANCE inside TEMPLATES
- VO-DML Attribute-s can have cardinality >1. Each value-type instance (*not an INSTANCE*), contained in single ATTRIBUTE with corresponding @dmrole
  - Simpler to interpret and parse, arguably
- Maybe get rid of CONSTANT
  - replace with LITERAL where it occurred
  - PARAMref is ugly as well

TEMPLATES/INSTANCE/ATTRIBUTE

```
<INSTANCE dmtype="sample:catalog.LuminosityMeasurement" ID="T1622053320664_208">
    <ATTRIBUTE dmrole="sample:catalog.LuminosityMeasurement.type">
        <LITERAL dmtype="sample:catalog.LuminosityType" value="magnitude"/>
    </ATTRIBUTE>
    <ATTRIBUTE dmrole="sample:catalog.LuminosityMeasurement.value">
        <COLUMN dmtype="ivoa:RealQuantity" ref="T1622053320664_1_F13"/>
    </ATTRIBUTE>
    <ATTRIBUTE dmrole="sample:catalog.LuminosityMeasurement.error">
        <COLUMN dmtype="ivoa:RealQuantity" ref="T1622053320664_1_F14"/>
    </ATTRIBUTE>
    <REFERENCE dmrole="sample:catalog.LuminosityMeasurement.filter">
        <REMOTEREFERENCE>http://ivoa.net/dm-instances/PhotometryFilter/SDSS.z.xml</REMOTEREFER
    </REFERENCE>
</INSTANCE>
```

PRIMARYKEY

- Identifies an INSTANCE of an ObjectType
- Introduced for serializations, *not* an element of VO-DML!
  - Implied for ObjectType-s that their instances are identified by identifier
  - In mapping, each INSTANCE of ObjectType MAY have a PRIMARYKEY
  - Used to identify and possibly refer to the INSTANCEs in the serialization using FOREIGNKEY construct!
  - No need to define explicitly in data model
- Content: PKFIELD, 1..*
  - PKFIELD has same content as non-structured ATTRIBUTE
- Why 1..*?
  May need multiple columns to define PRIMARYKEY
- Why name PRIMARYKEY?
  - Inspired by ORM and maps well to TAP_SCHEMA
  - See section 4.10 from VOTable spec
  - see also FOREIGNKEY described in REFERENCE below.

VOTable 1.4, section 4.10

4.10 The Relational Context
With a simple naming convention, the GROUP element may also specify some properties of the tables included in a VOTable document when a TABLE is viewed as a relation (part of a a relational database):
• A GROUP element having the name="primaryKey" attribute defines the primary key of the relation by enumerating the ordered list of FIELDrefs that make up the primary key of the table;
• A GROUP element having the name="foreignKey" attribute, with a ref="table reference" reference of the table having the associated primary ley, similarly enumerates the FIELDrefs of the foreign key;
• A GROUP element having the name="order" attribute may specify how the data are ordered.

TEMPLATES/INSTANCE with PRIMARYKEY

```
<INSTANCE dmtype="sample:catalog.Source" ID="T1622053320664_38">
    <PRIMARYKEY>
        <PKFIELD>
            <COLUMN dmtype="ivoa:string" ref="T1622053320664_1_F0"/>
        </PKFIELD>
    </PRIMARYKEY>
    <ATTRIBUTE dmrole="sample:catalog.AbstractSource.classification">
        <COLUMN dmtype="sample:catalog.SourceClassification" ref="T1622053320664_1_F15"/>
    </ATTRIBUTE>
    <ATTRIBUTE dmrole="sample:catalog.AbstractSource.position">
        <INSTANCE dmtype="sample:catalog.SkyCoordinate">
            <ATTRIBUTE dmrole="sample:catalog.SkyCoordinate.longitude">
                <COLUMN dmtype="ivoa:RealQuantity" ref="T1622053320664_1_F1"/>
            </ATTRIBUTE>
            <ATTRIBUTE dmrole="sample:catalog.SkyCoordinate.latitude">
                <COLUMN dmtype="ivoa:RealQuantity" ref="T1622053320664_1_F2"/>
            </ATTRIBUTE>
            <REFERENCE dmrole="sample:catalog.SkyCoordinate.frame">
                <REMOTEREFERENCE>http://ivoa.net/dm-instances/STC/Coords/ICRS.xml</REMOTEREFERENCE>
            </REFERENCE>
        </INSTANCE>
    </ATTRIBUTE>
```

OPTIONMAPPING

```
<COLUMN
 @dmtype
 @ref >
⊕ <OPTIONMAPPING>
</COLUMN>
```

```
<CONSTANT
 @dmtype
 @ref >
⊕ <OPTIONMAPPING>
</CONSTANT>
```

```
<LITERAL
 @dmtype
 @value
 @unit >
⊕ <OPTIONMAPPING>
</LITERAL>
```

- VOTable allows definition of list of possible values as OPTIONs on FIELD or PARAM
- VO-DML defines Enumerations and also allows assignment of "semantic vocabularies" to Attributes
- OPTIONMAPPING allows the mapper to define a translation from values in VOTable to corresponding valid values in the VO-DML Attribute it is assigned to.
  - Defined where the mapping between COLUMN and FIELD is defined
- Does LITERAL need it?
  - PARAM has OPTIONs doesn't it?
- (no example here)
- (may want something similar for mapping identifiers, or REMOTEREFERENCEs)

```
<OPTIONMAPPING>
 <MAPPEDOPTION>
( ENUMLITERAL | SEMANTICCONCEPT>)
</OPTIONMAPPING>
```

COMPOSITION

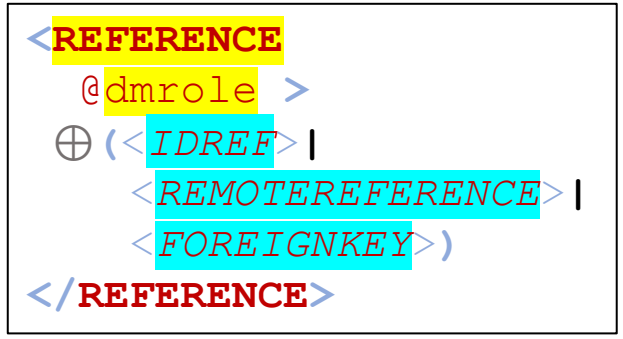
```
<COMPOSITION
  @dmrole >
 ⊕ <INSTANCE>
 ⊕ <EXTINSTANCES>
</COMPOSITION>
```

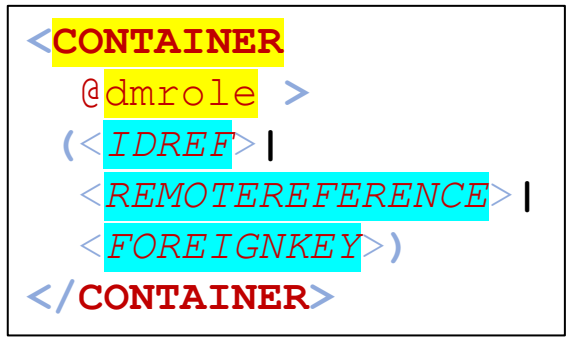
- Serialization of VO-DML's Composition, a parent-child relation between ObjectTypes, to a parent-child relation between INSTANCES
- In GLOBALS, COMPOSITION is simply containment of child INSTANCEs in parent INSTANCE.
  - All children for given Composition in *same* COMPOSITION element
- In TEMPLATES, child INSTANCEs in COMPOSITION can be in same row as parent INSTANCE
- Pure ORM represents this by two tables, with FOREGNKEY from child → parent!
  - Child INSTANCEs in COMPOSITION can be in child table, identifying their parent INSTANCE using a CONTAINER pointer
- Why named COMPOSITION (iso say COLLECTION as we used early on)
  - To be 1-1 with VO-DML
- (skip EXTINSTANCES, link to INSTANCE annotation on table that contains children of singleton instance. Inverse of CONTAINER)

GLOBALS/INSTANCE/COMPOSITION

```
<INSTANCE dmtype="coords:SphericalCoordSpace" ID="T1616959090495_71">
    <COMPOSITION dmrole="coords:CoordSpace.axis">
        <INSTANCE dmtype="coords:ContinuousAxis" ID="T1616959090495_86">
            <ATTRIBUTE dmrole="coords:Axis.name">
                <LITERAL dmtype="ivoa:string" value="ra"/>
            </ATTRIBUTE>
        </INSTANCE>
        <INSTANCE dmtype="coords:ContinuousAxis" ID="T1616959090495_107">
            <ATTRIBUTE dmrole="coords:Axis.name">
                <LITERAL dmtype="ivoa:string" value="dec"/>
            </ATTRIBUTE>
        </INSTANCE>
        <INSTANCE dmtype="coords:ContinuousAxis" ID="T1616959090495_128">
            <ATTRIBUTE dmrole="coords:Axis.name">
                <LITERAL dmtype="ivoa:string" value="r"/>
            </ATTRIBUTE>
        </INSTANCE>
    </COMPOSITION>
</INSTANCE>
```

REFERENCE (and CONTAINER)





- A VO-DML/Reference represents a(n often shared) *usage* of another ObjectType by an ObjectType or DataType
  - Often provides reference data to interprete the referencing object
  - E.f. reference frame provides meaning to values in a coordinate type
- For serialization most complex feature
  - It breaks nice tree-like hierarchy of many serializations
  - But data models generally are graph-like
- MUST not be serialized using containment of referenced INSTANCE!
  - Needs indirect linking somehow
- In ORM natural mapping to FOREIGNKEY
  - No natural mapping in XML (xlink not quite the same)
  - Nor in VOTable
  - But see section 4.10 of VOTable standard

REFERENCE continued

```
<REFERENCE
  @dmrole >
 ⊕(<IDREF>|
    <REMOTEREFERENCE>|
    <FOREIGNKEY>)
</REFERENCE>
```

```
<FOREIGNKEY>
 ⊕ <PKFIELD>
   <TARGETID>
</FOREIGNKEY>
```

```
<CONTAINER
  @dmrole >
 (<IDREF>|
  <REMOTEREFERENCE>|
  <FOREIGNKEY>)
</CONTAINER>
```

- Can be represented by IDREF to ObjectType INSTANCE in GLOBALS
- REMOTEREFERENCE represents the case where the INSTANCE has been defined inside some reference document.
  - E.g. all photometry filters, space coordinate systems etc could be in some registry?
  - REMOTEREFERENCE is an anyURI identifying such an INSTANCE
- FOREIGNKEY is a relation from INSTANCE serialized in one table to INSTANCE in another table
  - Note, the TARGETID identifies the container *INSTANCE*, NOT the table it is contained in !! Can have multiple INSTANCES, each with their own PRIMARY key serialized in some table
  - The referenced INSTANCE MUST have a PRIMARYKEY defined
  - The number and order of the PKFIELDs must match
- CONTAINER is a REFERENCE (of cardinality 1) from a child to a parent in an ORM-mapped parent-child composition relation
  - IDREF can be used if there is a singleton parent in GLOBALS
  - FOREIGNKEY for two tables
  - REMOTEREFERENCE useful?
  - @dmrole SHOULD identify the composition relation itself
    - (does VODMLMapper do this already?)

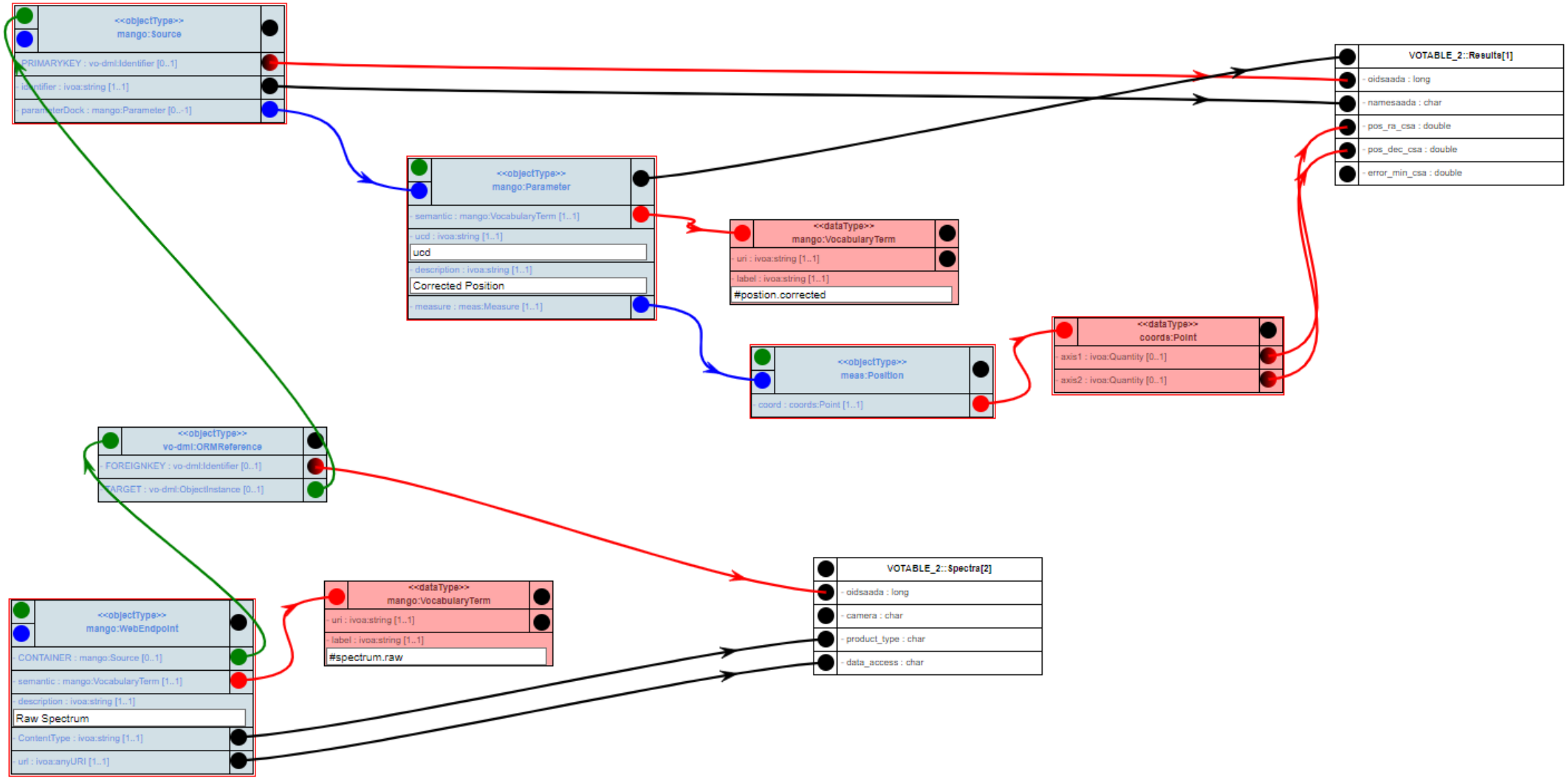
REFERENCE/IDREF

```
<GLOBALS>
    <INSTANCE dmtype="mango:extcoords.PhotometryCoordSys" ID="T1622165469013_230">
        <COMPOSITION dmrole="coords:PhysicalCoordSys.frame">
            <INSTANCE dmtype="mango:extcoords.PhotFilter" ID="T1622165469013_247">
                <ATTRIBUTE dmrole="mango:extcoords.PhotFilter.name">
                    <LITERAL dmtype="ivoa:string" value="sdss:u"/>
                </ATTRIBUTE>
            </INSTANCE>
        </COMPOSITION>
    </INSTANCE>
...
<TEMPLATES> ...
<INSTANCE dmtype="mango:extmeas.Photometry" ID="T1622165469013_175">
    <ATTRIBUTE dmrole="mango:extmeas.Photometry.coord">
        <INSTANCE dmtype="mango:extcoords.PhotometryCoord">
            <ATTRIBUTE dmrole="mango:extcoords.PhotometryCoord.luminosity">
                <COLUMN dmtype="ivoa:real" ref="T1622165469013_1_F5"/>
            </ATTRIBUTE>
            <REFERENCE dmrole="coords:Coordinate.coordSys">
                <IDREF>T1622165469013_230</IDREF>
            </REFERENCE>
        </INSTANCE>
    </ATTRIBUTE>
```

REFERENCE/REMOTEREFERENCE

```
<INSTANCE dmtype="meas:Position" ID="T1622165469013_80">
    <ATTRIBUTE dmrole="meas:Position.coord">
        <INSTANCE dmtype="coords:Point">
            <ATTRIBUTE dmrole="coords:Point.axis1">
                <COLUMN dmtype="ivoa:Quantity" ref="T1622165469013_1_F1"/>
            </ATTRIBUTE>
            <ATTRIBUTE dmrole="coords:Point.axis2">
                <COLUMN dmtype="ivoa:Quantity" ref="T1622165469013_1_F2"/>
            </ATTRIBUTE>
            <REFERENCE dmrole="coords:Coordinate.coordSys">
                <REMOTEREFERENCE>http://ivoa.net/dm-instances/STC/Coords/ICRS.xml</REMOTEREFERENCE>
            </REFERENCE>
        </INSTANCE>
    </ATTRIBUTE>
```

CONTAINER/FOREIGNKEY in VO-DML Mapper dm-usecases: combined_data



CONTAINER/FOREIGNKEY

```
<TEMPLATES tableref="T1616278954336_1">
    <INSTANCE dmtype="mango:Source" ID="T1616278954336_25">
        <PRIMARYKEY>
            <PKFIELD>
                <COLUMN dmtype="ivoa:string" ref="oidsaada_100"/>
            </PKFIELD>
        </PRIMARYKEY>
...

<TEMPLATES tableref="T1616278954336_14">
    <INSTANCE dmtype="mango:WebEndpoint" ID="T1616278954336_44">
        <CONTAINER>
            <FOREIGNKEY>
                <PKFIELD>
                    <COLUMN dmtype="ivoa:string" ref="oidsaada_199"/>
                </PKFIELD>
                <TARGETID>T1616278954336_25</TARGETID>
            </FOREIGNKEY>
        </CONTAINER>
...
```

Possible Adjustments

Omar Laurino has made suggestions for “simplifying the mapping syntax” in
https://cdn.rawgit.com/olaurino/presentations/6f7e5fdd/ivoa/victoria2018/slides.html

SIMPLIFYING THE MAPPING SYNTAX

View it on Github (https://github.com/olaurino/rama/tree/simplified-syntax)
[GL: seem to be based on <element> Definitions. would be good to have similar list for replacing type definitions. Are they also changed?]

ad5eb73 ATTRIBUTE -> ROLE
    [GL: also remove VODMLAttribute in favour of VODMLRole?]
f010d19 remove CONSTANT for INSTANCE
92ad210 support CONSTANT
61c6cf4 replace COLUMN with INSTANCE
3615fcc replace LITERAL with INSTANCE
    [GL: what would type of INSTANCE be? Also just the text? Complicates definition of INSTANCE]
a7a84bc use only role's name
    [GL: iso vodml-ref? for readability, IF vodml-id is generated, the name is part of the id anyway.
    For retrieval: simpler to create lookup table based on vodml-refs]
00c41d8 remove COMPOSITION and REFERENCE
    [GL: use ROLE instead? How enforce Schema-based validation different possible content?]

Omar’s RECOMMENDATION

- Explicit syntax is trivial-to-simple to implement for single table cases
- Maybe replace COMPOSITION, REFERENCE, ATTRIBUTE with ROLE
- Using attribute names imho only facilitates dynamic parsers
- Merging CONSTANT and COLUMN is a mess (compelling use case?)

Tricky features, maybe indicating missing features in vodml mapping:

CONTAINER can identify the composition relation it is a part of using @dmrole.
Q: Could/should we allow defining the composition from the parent down to the child mapping as well? Can do from GLOBAL ObjectType to a TEMPLATE: EXTINSTANCES

REMOTEREFERENCE must be an anyURI, i.e. no mapping possible say to a column identifying e.g. filet (Omar's example), possibly with a value mapping to URIs.
Note, VODML-Mapper does support linking REMOTEREFERENCE to column, but does not generate code for that.
Would likely need something like an OPTIONMAPPING, but then from string to URIs.

Should use REMOTEREFERENCE way more: requires creation of standard instance documents in firm location.

Can we extend ORMapping pattern with table column identifying objects defined in GLOBALS section? The @dmrole would help to select out the types that contain the specified Composition.

How to do inheritance mapping? E.g. class mapped over multiple tables in typical ORM pattern (e.g. JPA). Currently using SAMEINSTANCE reference for that.

Missing values mapping, e.g. -9999 to NULL. E.g. in sdss:DR15.Photoobjall

Thanks