Serializing VO-DML

Trying to be faithful.

The trouble with references and reference data

Tl;dr

- Standard serialization formats of VO-DML data models may be useful
  - They can be generated
- Especially for “reference data”
  - Coordinate frames
  - Photometry filters
  - SimDM Protocols
  - etc
- Need way to reference/refer to these from other serializations
  - If the reference itself is human readable, nice.

What does one do with data models?

Create *instances*

What are data model *instances*?

Think of Objects for Classes

How does one create instances?

Using a *serialization*

What is a ***serialization***?

Representation of *instances* of (parts of) a data model in computer(human?) readable form

What we do with instances depends on representation

- XML (JSON, YAML, ..): write them, send them, upload them
- RDB: insert them in DB, query DB
- Java (C#,python, …): manipulate them, validate them, mediate them

Some early work on formalizing data model serialization in the VO

- 2004 Cambridge: “Unified domain model for Astronomy”
- 2005 Kyoto: discussing registry and VOTable from UML perspective
  - VOTable still has/d remnants in XSD comments
- 2008 Trieste: VO-URP (with Laurent Bourges)
- 2012 SimDM spec
- 2012 UTYPES Tiger team
- 2013->? Mapping spec
  - Various tools
- 2018 VO-DML spec
- 2017,2018:
  - Baptiste Cecconi asks for XSD from VO-DML for STC
  - Tim Jenness asks (Omar) for YAML/JSON serialization from VO-DML

VOTable mapping spec

- Note: Complex models require complex mapping, in any syntax.
  - See last interop
- First apply it to simpler models
  - STC?
- Should we go back to original proposal: GROUP + UTYPE?
  - Or can we use current, comprehensive proposal as guide to create simpler version?

Problem: Not *faithful*

A VOTable in general is a representation of a transformed, alternative version of some *ad hoc* model that uses part of original model.

Faithful serializations

1-1

Question:

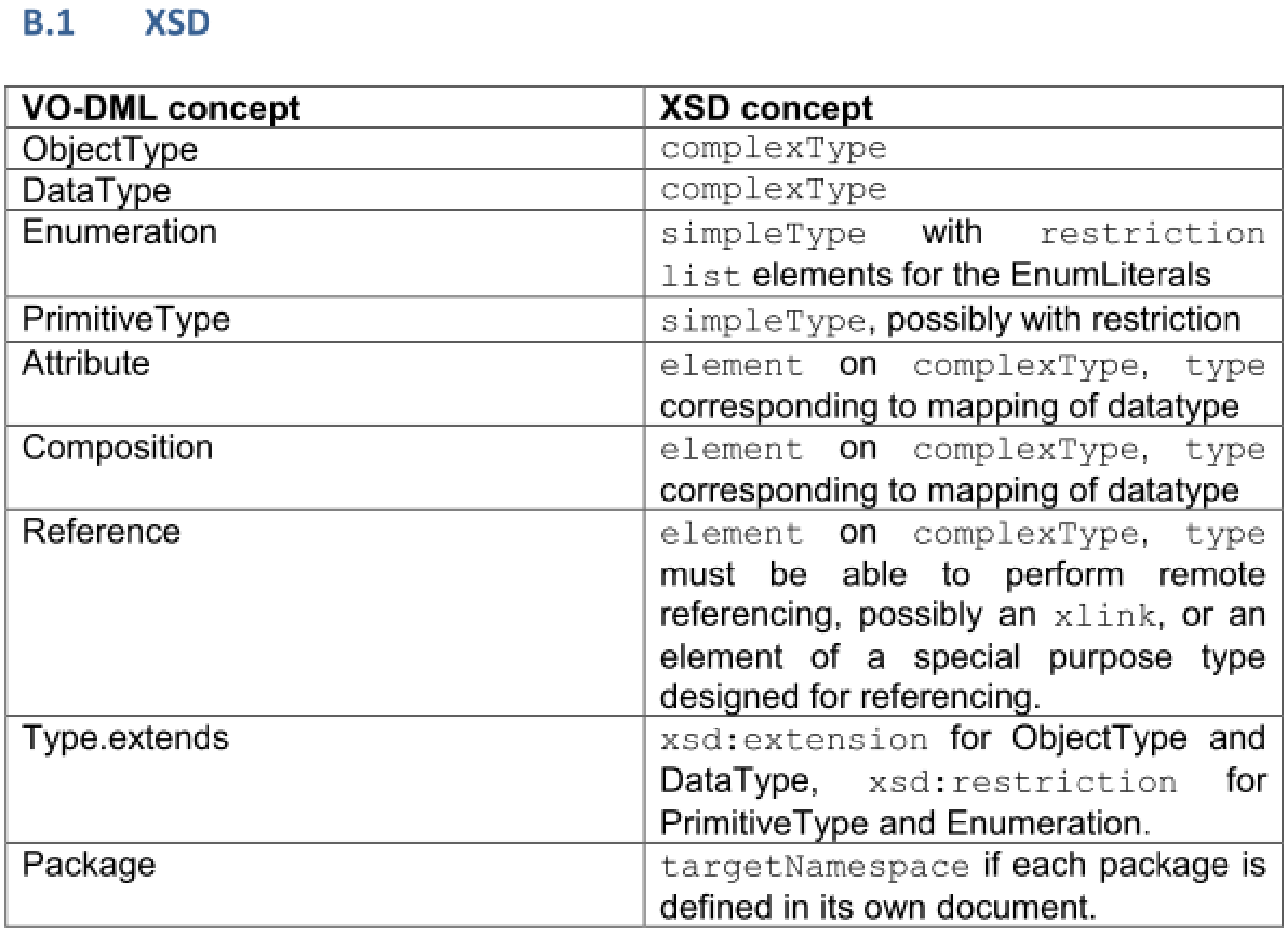
Can standard serialization(format)s be generated from VO-DML?

Answer:

Yes: VO-URP/SimDM (DDL, XSD, Java)
See: some scripts in <volute>/vo-dml/tools/xslt (HTML, Java, XSD)

Focus on XSD

VO-DML spec Appendix B



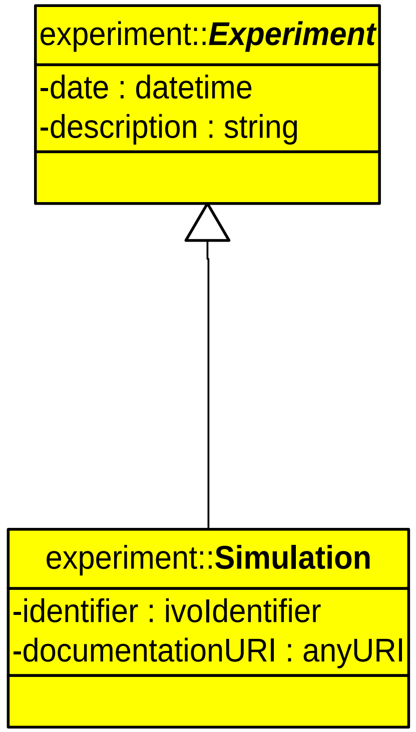
B.1 XSD

VO-DML concept	XSD concept
ObjectType	`complexType`
DataType	`complexType`
Enumeration	`simpleType` with `restriction list` elements for the EnumLiterals
PrimitiveType	`simpleType`, possibly with restriction
Attribute	`element` on `complexType`, `type` corresponding to mapping of datatype
Composition	`element` on `complexType`, `type` corresponding to mapping of datatype
Reference	`element` on `complexType`, `type` must be able to perform remote referencing, possibly an `xlink`, or an element of a special purpose type designed for referencing.
Type.extends	`xsd:extension` for ObjectType and DataType, `xsd:restriction` for PrimitiveType and Enumeration.
Package	`targetNamespace` if each package is defined in its own document.

Model import TBD

Example: Mapping Object Types

- Every class -> globally defined complexType
- isAbstract -> abstract="true"
- Attribute -> element of simplish-type, either built-in, or also generated


experiment::*Experiment*
-date : datetime
-description : string
experiment::**Simulation**
-identifier : ivoIdentifier
-documentationURI : anyURI

```
<xsd:complexType name=„Experiment" abstract="true">
  <xsd:complexContent>
    <xsd:extension base="base:DataObject">
      <xsd:sequence>
        <xsd:element name="date" type="xsd:datetime"/>
        <xsd:element name="description" type="xsd:string"/>
          ...
        </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

Automated Generation

- XSLT script: vo-dml2xsd
    - http://volute.g-vo.org/svn/trunk/projects/dm/vo-dml/tools/xslt/vo-dml2xsd.xsl
- Differences wrt VO-URP
    - Whole model in one XSD, iso one file per package
    - Need to deal with model imports
    - **Needs review: is this what you/we want?**
- Examples
    - SimDM-v1.1
    - STC (MC-D)

JSON schema from XSD

- JSON Schema: http://json-schema.org
- Want a vo-dml2jsonschema.xsl ???
- From XSD: https://github.com/highsource/jsonix-schema-compiler/wiki/Turning-on-JSON-Schema-Generation
    - SimDM 1.1: http://volute.g-vo.org/svn/trunk/projects/theory/snapdm/specification/v1.1/vo-dml/xsd

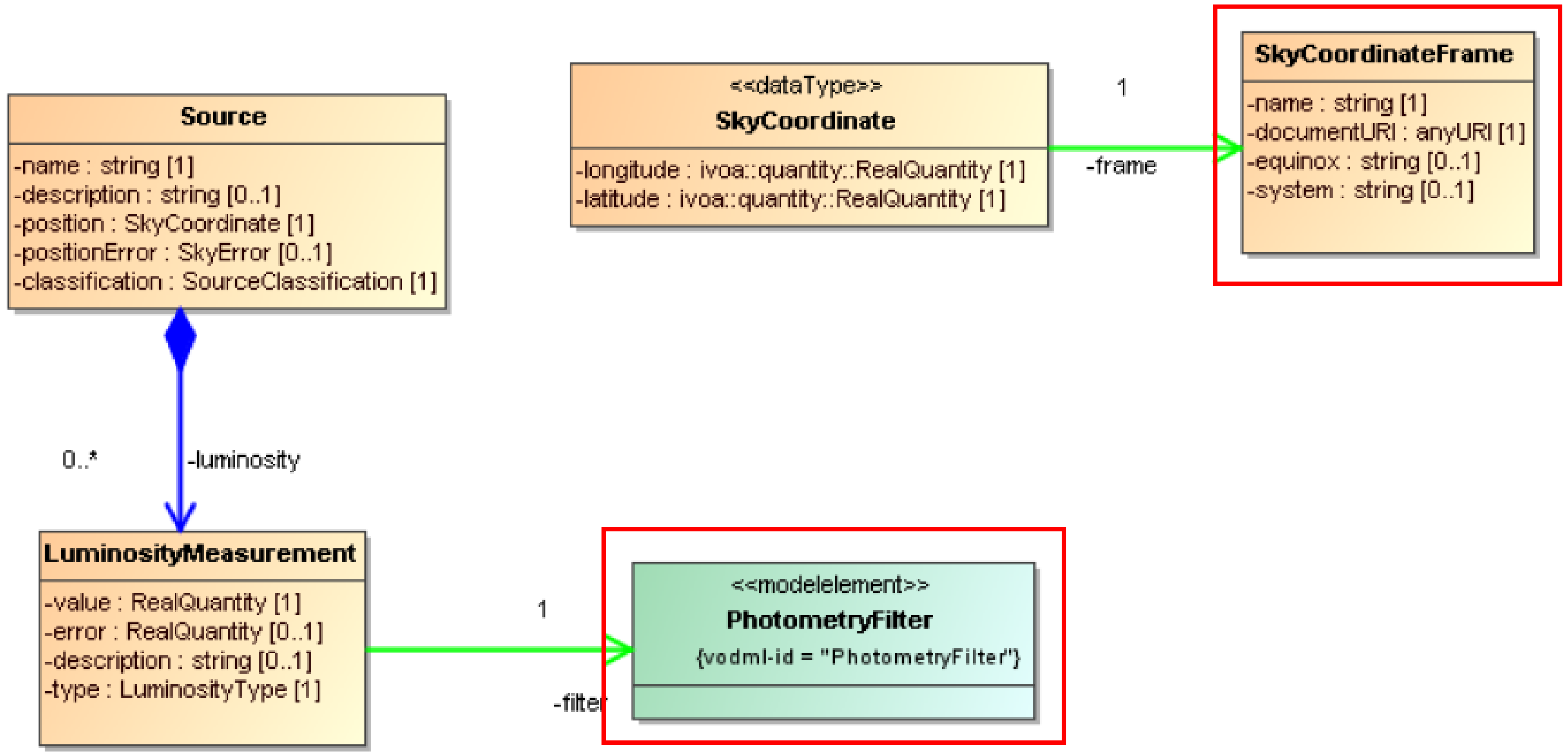
YAML

- I don’t know YAML
- May it use JSON schema? https://json-schema-everywhere.github.io/yaml

References, reference data

- XML inherently tree-like, not graph-like ala RDB
- Need to deal with many-to-1 references
    - Different form composition!
- References to model instances *outside* the current document
    - Maybe different representation?
- In XML, can Xlink be used ?
    - Not quite same semantics
- Curation?
- Registration?

Reference data: simple TOY source model


Source
-name : string [1]
-description : string [0..1]
-position : SkyCoordinate [1]
-positionError : SkyError [0..1]
-classification : SourceClassification [1]
<<dataType>>
SkyCoordinate
-longitude : ivoa::quantity::RealQuantity [1]
-latitude : ivoa::quantity::RealQuantity [1]
1
-frame
SkyCoordinateFrame
-name : string [1]
-documentURI : anyURI [1]
-equinox : string [0..1]
-system : string [0..1]
0..*
-luminosity
LuminosityMeasurement
-value : RealQuantity [1]
-error : RealQuantity [0..1]
-description : string [0..1]
-type : LuminosityType [1]
1
-filter
<<modelelement>>
PhotometryFilter
{vodml-id = "PhotometryFilter"}

Question:

Do we want to standardize VO-DML serialization formats?