The University of Manchester
Jodrell Bank
Observatory

VO-DML Extensions

Paul Harrison (JBO)
IVOA Interop Spring 2024

Introduction

- VO-DML Tooling update introduced in previous Interop talks now quite mature.
  - Refined by the needs of ProposalDM, the generated code for which is used as the serialisation basis for Polaris, a proposal submission toolkit.
  - Has already introduced some extensions to VO-DML that have not yet been included in the standard document.
- This talk
  - Updates on the VO-DML tooling (since Bologna)
  - Suggestions for VO-DML 1.1 WD
  - Thoughts on the modularity of existing DMs.

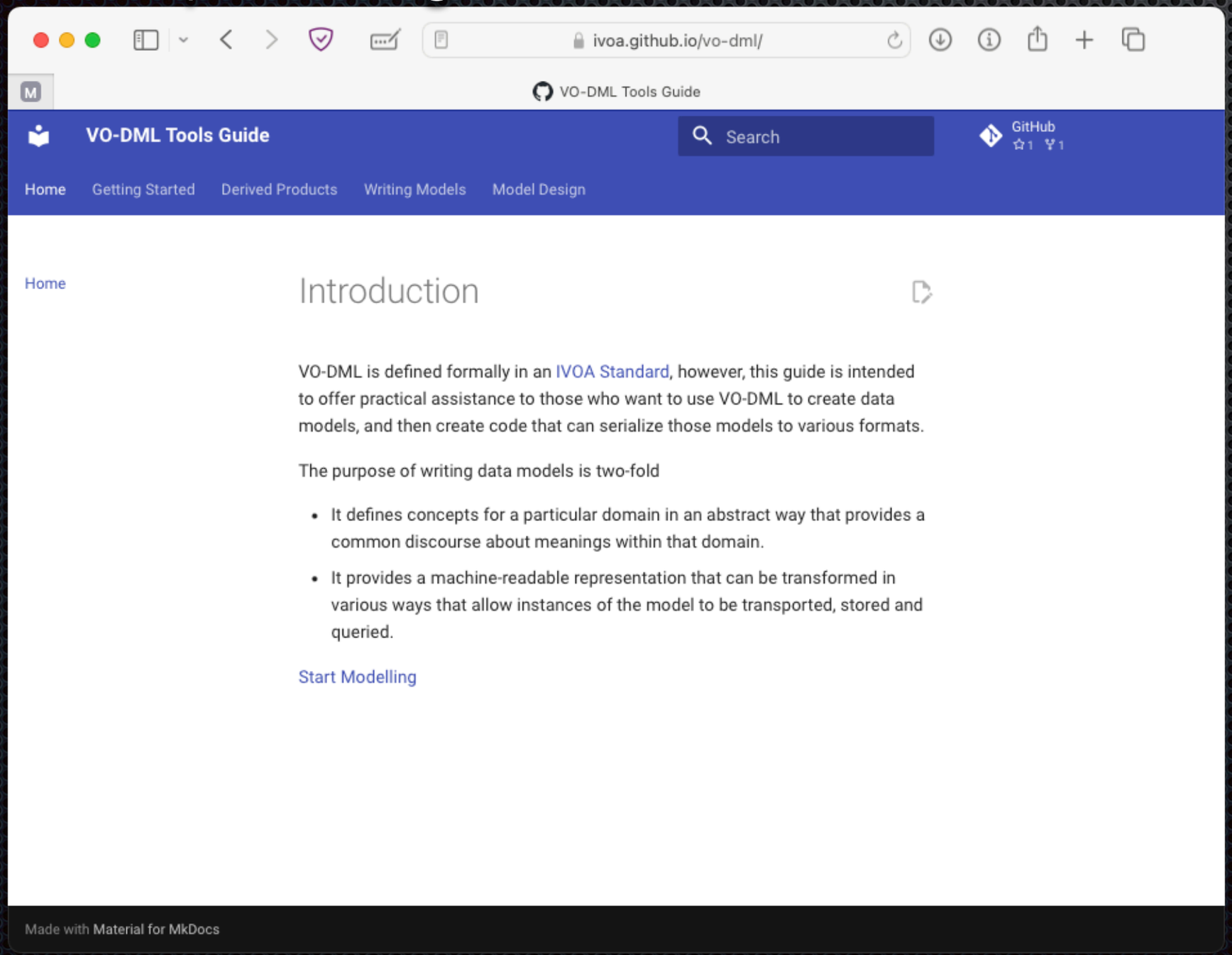
VO-DML Tooling Updates

- Significant updates since a year ago (v0.3.19) when last reported - now v0.5.1
  - Tools documentation
  - Improved generated model documentation
  - Contained references support in Java.
  - XML and JSON schema generation.

The University of Manchester
Jodrell Bank
Observatory

Tool Documentation

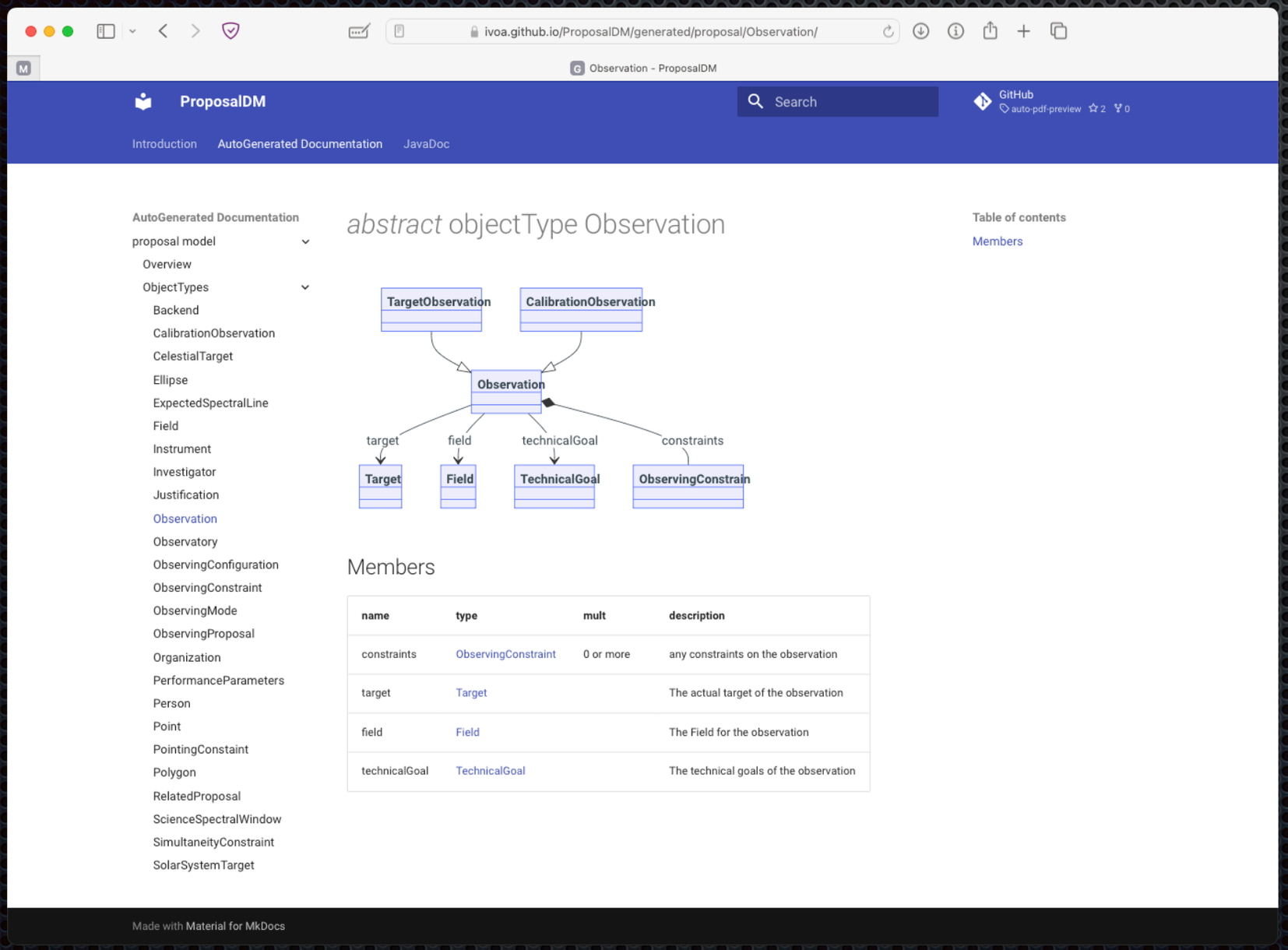
- https://ivoa.github.io/vo-dml/



The University of Manchester
Jodrell Bank Observatory

Model Site Documentation

e.g. ProposalDM



- individual pages for each model element
- neighbourhood diagram
- uses mkdocs

VO-DML 1.1 WD

- Backwards compatible extensions (as required)
  - already tested in deployed tools plugin
- Managed via GitHub milestones with PR for each feature
- Main update for 1.1 on the 20-update-vo-dml-standard-document branch
- Original 1.0 REC was written in Word - the 1.1 WD is in markdown (via an automated conversion with pandoc)
  - might even produce yet another publishing option via pandoc

VODML-ID syntax made normative

- In the VO-DML meta-model XML schema VODML-ID is simply a string, rather than an ID/IDREF structure, so having arbitrary form would be potentially problematic as there would be no validation via the schema - although the standard says that they should be unique.
  - Data models that were created via the original tooling have the (proposed) normative form anyway as the UML to VO-DML conversion generated such elements.
- Originally the textual syntax of the VODML-ID for each model element was only specified in an appendix - moved to main body to become normative
  - essentially the VODML-ID is derived from the location in the model
- Tooling now checks that VODML-ID is correct via a schematron rule, however
  - tooling never “reads” that element value - it always “calculates” it, so the element could be removed from VO-DML schema

VO-DML extension - Natural Keys

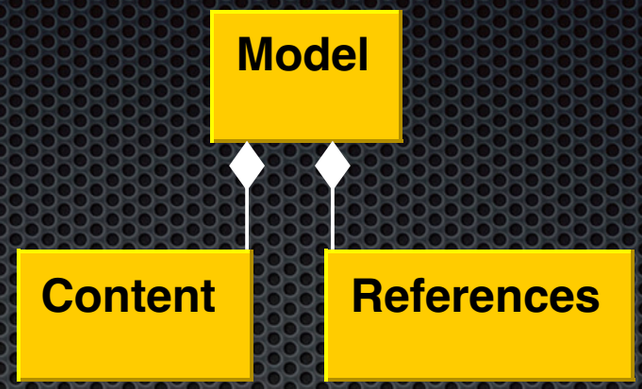
- Object Relational Mapping uses surrogate keys widely - however, in the model it is sometimes better to use a “natural key” i.e. an existing attribute - often the case for the target of “references”.

```
<xsd:complexType name="NaturalKey">
    <xsd:annotation>
      <xsd:documentation>
        This constraint is used to indicate that an attribute is a natural key for its owning ObjectType, meaning that the
        attribute value should be globally unique. This may be applied multiple times to indicate that only a composition
        of several attributes make the globally unique key.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexContent>
      <xsd:extension base="Constraint">
        <xsd:sequence>
          <xsd:element name="Position" type="xsd:positiveInteger">
                <xsd:annotation>
                  <xsd:documentation>In the case where multiple attribute values make up the natural key, this
                  value indicates the ordinal number of this particular key in the compound key.</xsd:documentation>
                </xsd:annotation>
          </xsd:element>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
```

VO-DML Metamodel XML Schema updates

- aforementioned natural key extension
- make <name> and <documentationURL> optional (and deprecated) in the <import> as they merely repeat information that is in the imported document
- Has already happened on the main branch - non-breaking - following the XML schema versioning endorsed note.

Serialisation



- Appendix B in the 1.0 document describes how the model *might* be serialised
- Current tooling attempts to produce a **standard** serialisation for XML and JSON
  - based on the UML above so that a single model instance serialisation will contain both the content and references
    - references that are not otherwise “contained” (see later) are emitted in the references section
  - tooling creates both XML and JSON schema which can be used to validate model instances.
- Proposal is to rewrite Appendix B to make clear that new serialisation is intended for interoperability, and thus “standard”.
- Note that this form of serialisation is more suitable for writing REST web service interfaces for the models than MIVOT.

Serialization 2

XML

```xml
<ser:myModelModel xmlns:ser="http://ivoa.net/vodml/sample/serialization" >
    <refs>
        <refa _id="id_0">
            <val>a value</val>
        </refa>
        <refb>
            <name>a name</name>
            <val>another val</val>
        </refb>
    </refs>
    <SomeContent>
        <zval>a z val</zval>
        <ref1>id_0</ref1>
        <ref2>a name</ref2>
    </SomeContent>
</ser:myModelModel>
```

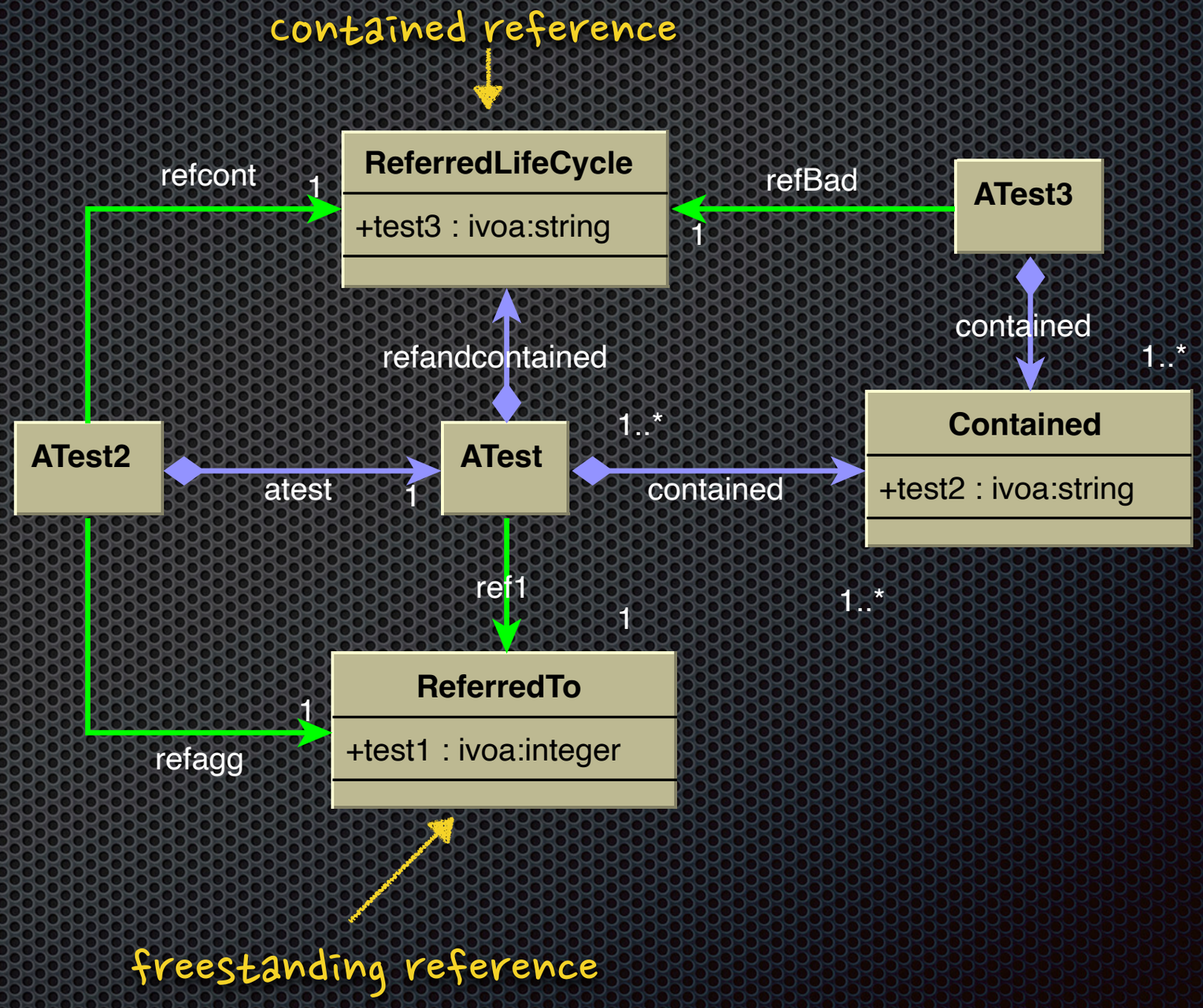
JSON

```json
{
  "MyModelModel" : {
    "refs" : {
      "MyModel:Refa" : [ {
        "_id" : 0,
        "val" : "a value"
      } ],
      "MyModel:Refb" : [ {
        "name" : "a name",
        "val" : "another val"
      } ]
    },
    "content" : [ {
      "@type" : "MyModel:SomeContent",
      "zval" : "a z val",
      "ref1" : 0,
      "ref2" : "a name"
    } ]
  }
}
```

- https://ivoa.github.io/vo-dml/Serialization/
- note that tooling includes automated round-trip serialisation unit tests against generated schema.

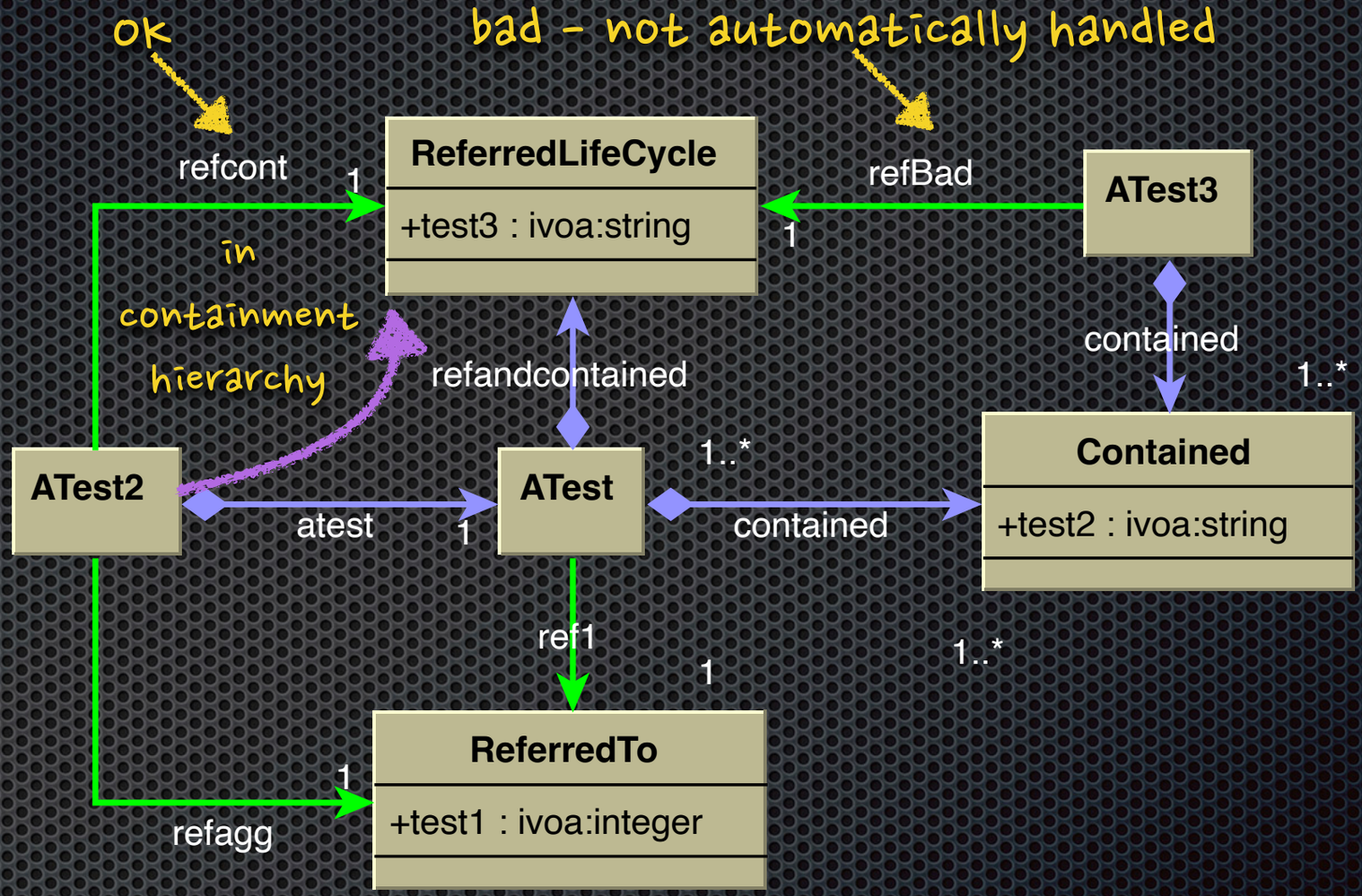
Reference Lifecycle/Containment

- Original tooling/std assumed that references "freestanding" - i.e. lifecycles independent
- In latest tooling references can be "contained" i.e. referenced element can exist as a composition within some parent.



Reference Lifecycle/Containment 2

- tooling will generate Java code that will deal properly with contained references
- schematron rules warn of "dangerous" contained reference use

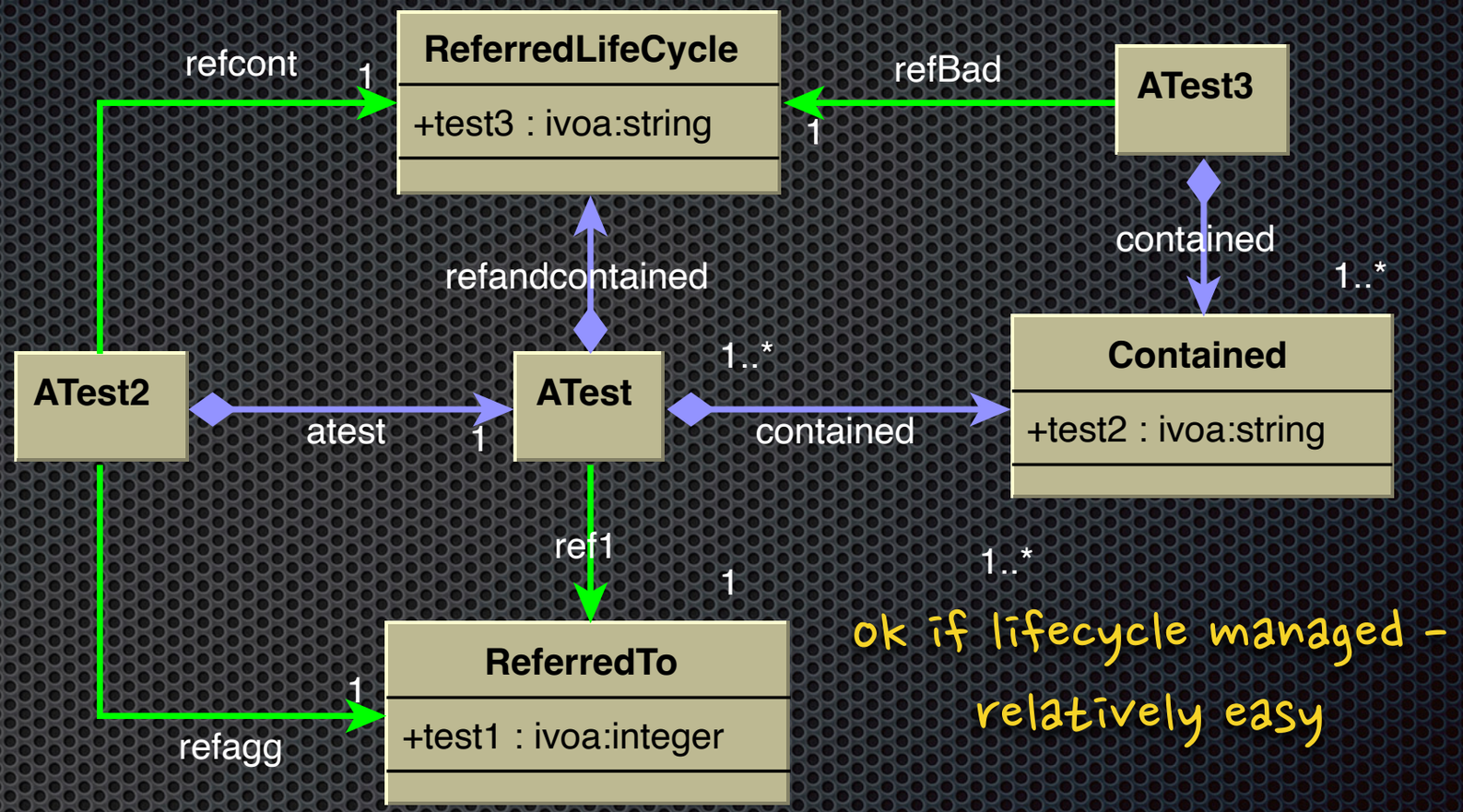



```
failed-assert /Q{http://www.ivoa.net/xml/VODML/v1}model[1]/Q{}
objectType[6]/Q{}reference[1]

Reference lifecycleTest:ReferredLifeCycle used in
ATest3.refBad is already use in unrelated composition ATest
which has lifecycle implications (i.e. the reference could
disappear unless code is aware of relationship)
```

Reference Lifecycle/Containment 3

- Schematron complains with "unique composition rule"
- however, this is just a warning
- Wording in Standard probably OK



ok if lifecycle managed - relatively easy

```
failed-assert /Q{http://www.ivoa.net/xml/VODML/v1}
model[1]/Q{}objectType[6]/Q{}composition[1]/Q{}
datatype[1]/Q{}vodml-ref[1]


objecttype lifecycleTest:Contained is used more than
once, as target of composition relation. In this case for
containing objectType lifecycleTest:ATest3


      ** (this message will repeat itself 2 times!,
once for each different container) **
```

IVOA Base Model Additions

- Following on from the serialisation and reference containment discussions it is useful to be able to mark in a model where the intention is to point to an external entity (which cannot be done with references as they are internal)

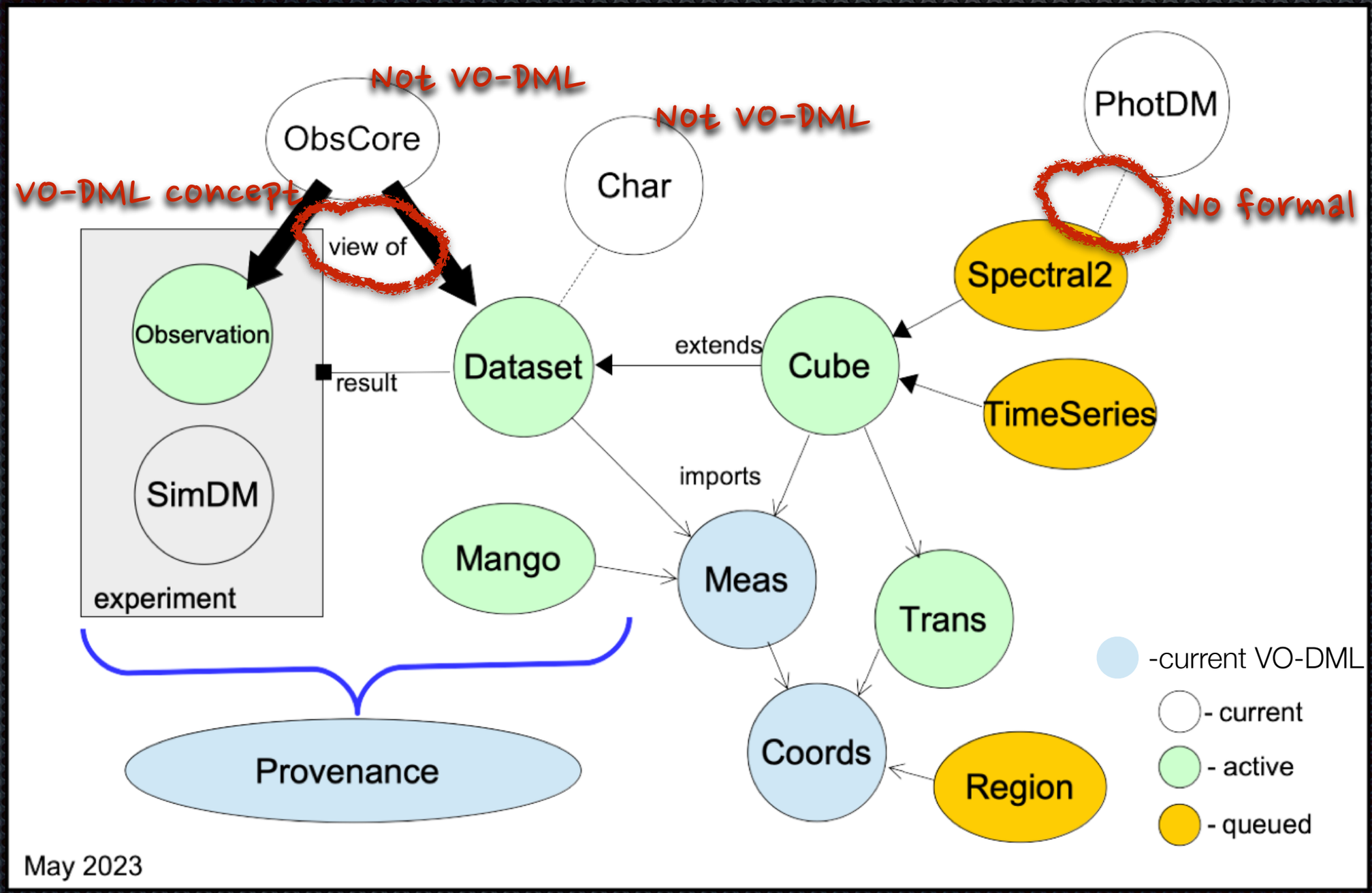
```
primitive intIdentifier -> integer "an integer identifier that can be used as a key for lookup of an entity that is *outside this datamodel*"
primitive stringIdentifier -> string "a string identifier that can be used as a key for lookup of an entity that is *outside this datamodel*"
primitive ivorn -> anyURI "an identifier that can be used as a key to look up in an IVOA registry - see https://www.ivoa.net/documents/IVOAIdentifiers/"
```

- This is done on the base_update branch

VO-DML 1.2 and beyond

- Lots of potential ideas/improvements, but have left them out of 1.1 in the hope of speeding up approval of this document.
  - specifying UCDs
    - could then automatically create TAP schema/services
  - concept of Choice/OneOf
  - some specific simple constraints
    - e.g greaterThan

DM landscape



- Taken from Mark C-D's talk last interop

Importance of VO-DML

- Provides rigour in the DM design
  - provides a framework for validating instances.
- Allows real re-use
- Can show up duplication more easily
- There are are growing number ObsCore extensions that do not have a VO-DML basis
  - and ObsCore is widely cited as a “view” on other DMs
  - perhaps we should try to formalise this “view” concept in VO-DML
  - or we have a DM that is not as “simple” as ObsCore

DM concept duplication

Concept	DM
Observatory (Facility)	VOResource, ProposalDM
Instrument	VODataService,ProposalDM,CAOM
Service	VOResource, Mango, Datalink
Data Set	DataSet,VODataService
Observation	ObsCore, ProposalDM
Creator/Curator	VOResource,DataSet
Parameter	PDL, VOTable, UWS, Mango

- Not guaranteed to be an accurate/complete survey, but illustrative
- Sometimes the concept might be intended as a reference
- Note that data models are created ‘outside’ the DM WG

Other approaches

- SPASE (Space Physics Archive Search and Extract)
- NASA PDS
- These are easier to query - the whole “Domain” is in the one model (or set of non-overlapping models)
  - No possible ambiguities in concepts
  - Don’t have to try to join together services as with the IVOA
    - we don’t have that luxury - we are a federation

DM rationalization/refactor

- I am trying to facilitate the possibility of trying out some refactoring in https://github.com/ivoa/DataModelPlayground
  - note that the volute repository did have more model representations http://volute.g-vo.org/svn/trunk/projects/dm/vo-dml-org/models/ - but this got sidelined
- Not trying to find a complete normalisation of all models, but it would be good to create a “basis set” with no duplication
  - could feed into the P3T efforts

Personal Vision

- ObsCore has no extensions - remains “core” to all wavelengths/data types - possibly some extra “columns” discovered from new domains - but definitely should not be sparse.
- Registry (or a slight extension of) stores definitive (i.e VO wide) instances of “slowly changing” model elements -e.g. Observatories, Telescopes and Instruments.
- New DM factoring of something at roughly the Characterization/Dataset level, but composed of more smaller reusable sub-models (e.g. CatalogueSource), to be basis of more sophisticated data discovery/manipulation service.
  - perhaps one input to this service could be the DataLink response from the ObsCore discovery