Mapping Syntax

Laurent Michel, Mark Cresitello-Dittmar et al.

https://github.com/ivoa-std/ModelInstanceInVot

Model Mapping in VOTable

- **DESIGN**
  - Convenient syntax being valid against the workshop use-cases
  - Smooth VOTable integration
  - Exercice a validation process

- **DELIVERABLES**
  - XML schema
  - IVOA document
  - Client tooling

VOTable Insertion

- **A strong demand for a shy integration**
  - Do not break working things
  - Do not bother existing VOTable stakeholders.

- **Encapsulating the mapping block into a <resource>**
  - The VOTABLE schema supports resources whose content is not controlled by the schema
    - `type=meta`
    - `ns=dm-mapping`

```
<!--
   Suggested Doug Tody, to include new RESOURCE types
-->
<xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
```

- **2 separate schemas**
  - The mapping schema has no connection with the VOTable schema
  - Mapped documents must be validated against both schema separately
  - To achieve this separation, we took care to use different terms for the content in order to not mislead legacy clients doing XPath-based parsing.

VOTable Insertion

- **Mapping scope limited to one resource**
  - Several result resources, each with its own mapping, can be stacked in one VOTable
  - Mapped data can be distributed over multiple tables

```
<?xml version="1.0" encoding="UTF-8"?>
<VOTABLE xmlns="http://www.ivoa.net/xml/VOTable/v1.3" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    version="1.3">

    <RESOURCE type="results">
        <RESOURCE type="meta">

            <dm-mapping:VODML xmlns:dm-mapping="http://www.ivoa.net/xml/merged-syntax">

        </RESOURCE>

        <TABLE name="Results">
            <PARAM ID="_title" name="title" value="TilteReadInParam" datatype="char" arraysize="*" />
            <FIELD ID="_poserr_148" name="oidsaada" datatype="long" ucd="meta.id;meta.main" />
            <DATA>
        </TABLE>

        <TABLE name="OtherResults">
        <TABLE name="Spectra">
    </RESOURCE>
</VOTABLE>
```

Syntax Overview

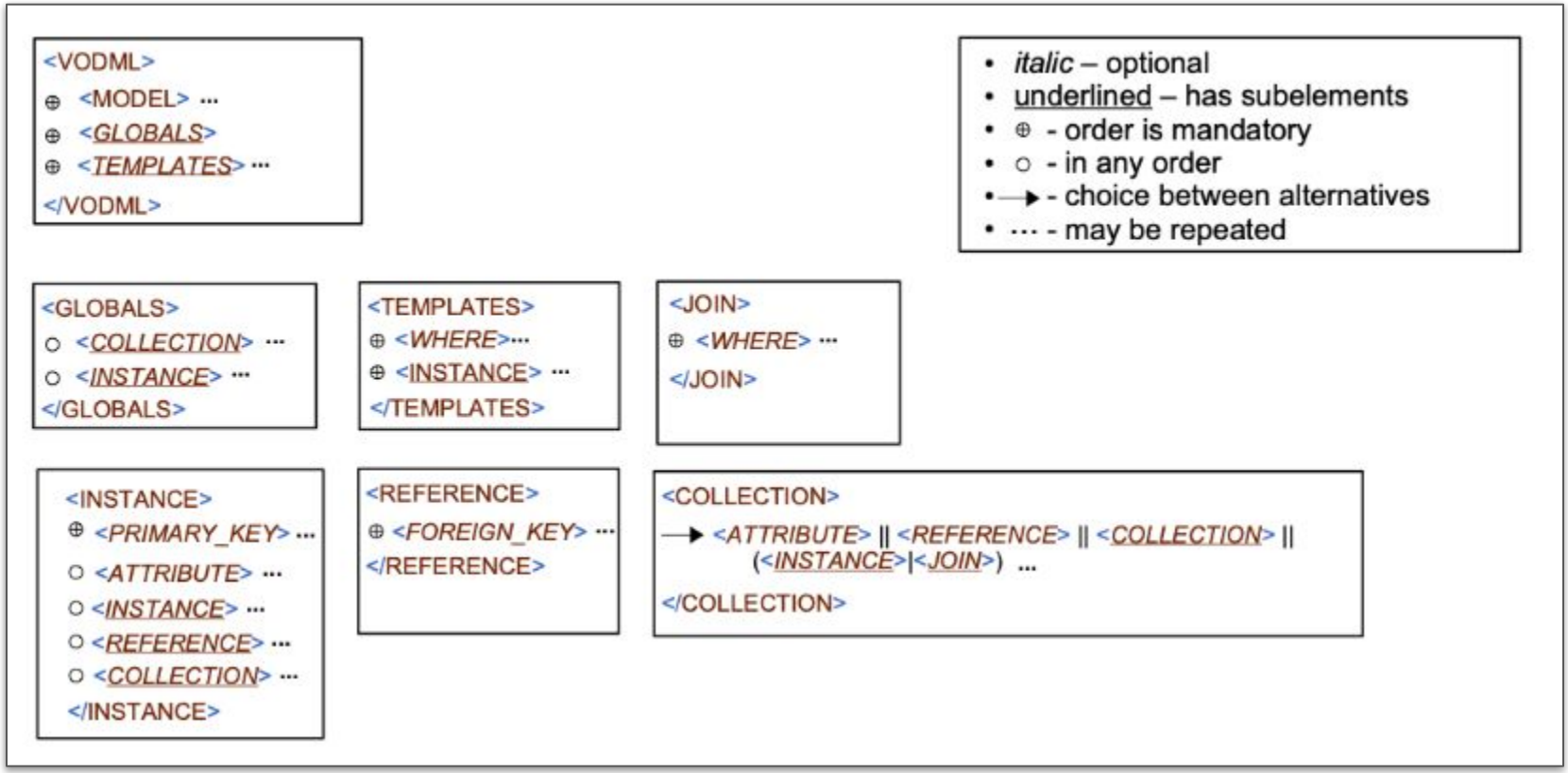
- **Mapping block structure**
  - One <VODML> block, the container
  - Followed but one <MODEL> element per used model
  - Followed by one <GLOBALS> element
    - Mapping of quantities that do not relate to table rows
  - Followed by one <TEMPLATES> element per mapped table
    - Container for the table row mapping

```
<dm-mapping:VODML xmlns:dm-mapping="http://www.ivoa.net/xml/merged-syntax">
    <dm-mapping:MODEL name="test" />

    <dm-mapping:GLOBALS>

    <dm-mapping:TEMPLATES tableref="Results">

    <dm-mapping:TEMPLATES tableref="OtherResults">

    <dm-mapping:TEMPLATES tableref="Spectra">
</dm-mapping:VODML>
```

Syntax Overview: XML elements

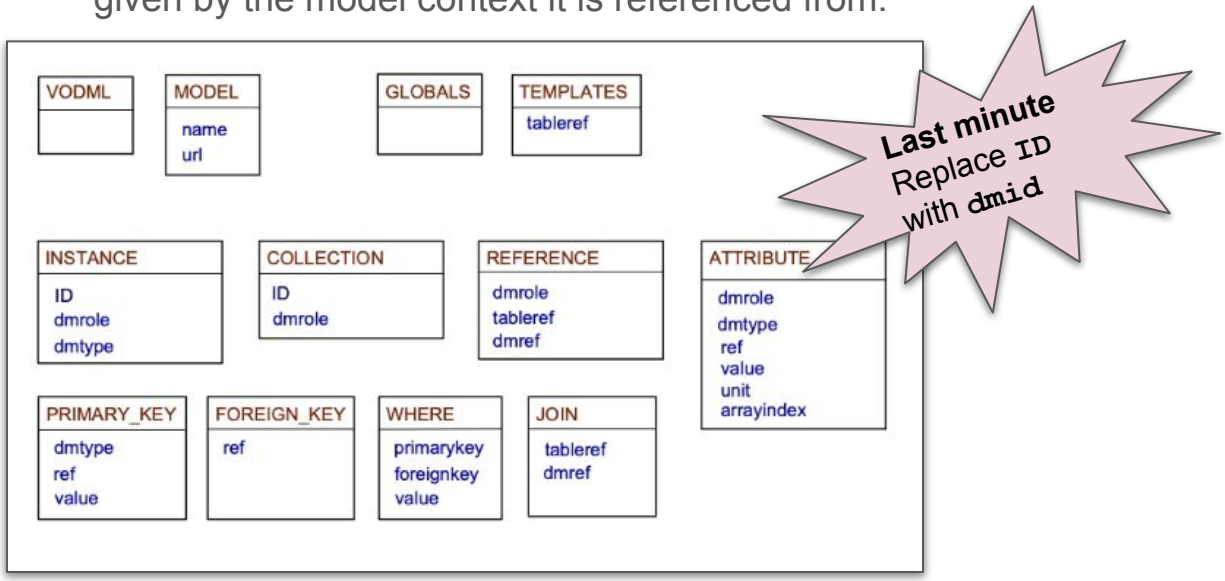
- **Mapping elements**
  - Mapping block containers
    - **<VODML>**, **<GLOBALS>** and **<TEMPLATES>**
  - Class hierarchy
    - **<INSTANCE>**: complex date type or object type
    - **<ATTRIBUTE>**: simple attribute (atomic value)
    - **<COLLECTION>**: composition of instances or array of attributes
  - Data relationships
    - **<REFERENCE>:** Instance reference
    - **<JOIN>:** Data join with either a **<COLLECTION>** or table rows
    - **<WHERE>: <JOIN>** condition or table row filter
    - **<PRIMARY_KEY>:** Reference to a value that can be used as primary key for either table rows or **<COLLECTION>** items.
    - **<FOREIGN_KEY> :**Foreign key to be used to resolve a **<REFERENCE>**

Syntax Overview: XML elements



Syntax Overview: Elements @attributes

- **Flexible @attributes roles**
  - The valid attribute patterns for a given element depend on the local context
    - An **<INSTANCE>** placed in **<GLOBALS>** has no specific role (no **@dmrole**), the role is given by the model context it is referenced from.



  - These rules are enforced by the schema

Example of @attribute patterns

Attribute set with *michel* as a literal value

```
<dm-mapping:ATTRIBUTE dmrole="test.owner.name" dmtype="string" value="Michel" />
```

Attribute set with the value read from the *_title* table column

```
<dm-mapping:ATTRIBUTE dmrole="test.title" dmtype="string" ref="_title" />
```

Attribute set with a value read in the *_title* if it exists or with a literal value

```
<dm-mapping:ATTRIBUTE dmrole="test.title" dmtype="string" ref="_title" value="default title"/>
```

Syntax Overview

- **A very simple object instance**

```
<dm-mapping:INSTANCE dmrole="coords:TimeFrame.refPosition" dmtype="coords:StdRefLocation">
  <dm-mapping:ATTRIBUTE dmrole="coords:StdRefLocation.position" dmtype="ivoa:string" value="BARYCENTER"/>
</dm-mapping:INSTANCE>
```

- **A more complex object instance**

```
<dm-mapping:INSTANCE dmid="_ts_data" dmrole="" dmtype="cube:NDPoint">
  <dm-mapping:COLLECTION dmrole="cube:NDPoint.observable">
    <dm-mapping:INSTANCE dmtype="cube:Observable">
      <dm-mapping:ATTRIBUTE dmrole="cube:DataAxis.dependent" dmtype="ivoa:boolean" value="False"/>
      <dm-mapping:INSTANCE dmrole="cube:MeasurementAxis.measure" dmtype="meas:Time">
        <dm-mapping:INSTANCE dmrole="meas:Time.coord" dmtype="coords:MJD">
          <dm-mapping:ATTRIBUTE dmrole="coords:MJD.date" dmtype="ivoa:real" ref="_obstime"/>
          <dm-mapping:REFERENCE dmrole="coords:Coordinate.coordSys" dmref="_timesys"/>
        </dm-mapping:INSTANCE>
      </dm-mapping:INSTANCE>
    </dm-mapping:INSTANCE>
    <dm-mapping:INSTANCE dmtype="cube:Observable">..
    <dm-mapping:INSTANCE dmtype="cube:Observable">..
  </dm-mapping:COLLECTION>
</dm-mapping:INSTANCE>
```

Syntax Overview: Reference

```
<dm-mapping:COLLECTION dmid="_Datasets" dmrole="">
  <dm-mapping:INSTANCE dmid="_ds1" dmrole="" dmtype="ds:experiment.ObsDataset">
   <dm-mapping:PRIMARY_KEY dmtype="ivoa:string" value="5813181197970338560"/>
    <dm-mapping:ATTRIBUTE dmrole="ds:dataset.Dataset.dataProductType" dmtype="ds:dataset.DataProductType" value="TIMESERIES"/>
    <dm-mapping:ATTRIBUTE dmrole="ds:dataset.Dataset.dataProductSubtype" dmtype="ivoa:string" value="GAIA Time Series"/>
    <dm-mapping:ATTRIBUTE dmrole="ds:experiment.ObsDataset.calibLevel" dmtype="ivoa:integer" value="1"/>
    <dm-mapping:REFERENCE dmrole="ds:experiment.ObsDataset.target" dmref="_tg1"/>
  </dm-mapping:INSTANCE>
</dm-mapping:COLLECTION>
<dm-mapping:INSTANCE dmid="_tg1" dmrole="" dmtype="ds:experiment.Target">
  <dm-mapping:ATTRIBUTE dmrole="ds:experiment.BaseTarget.name" dmtype="ivoa:string" value="5813181197970338560"/>
</dm-mapping:INSTANCE>
```

The **`ds:experiment.ObsDataset.target`** object will be set with **`ds.experiment.Target`** instance

Syntax Overview: Joins

- **Flexible JOIN operator**
  - Join 2 data collections
    - Static data or **<TABLE>** templates
  - Can join **<TABLE>** data together
  - Can join **<TABLE>** data with static data collections
  - Can join static data collection together
  - Support multiple join conditions (**<WHERE>**)

```
<dm-mapping:COLLECTION   dmid="id5" dmrole="">
        <!-- Test Case 9.6b: tableref + no dmref + WHERE (multiple) -->
        <dm-mapping:JOIN tableref="aaaaa" >
                <dm-mapping:WHERE foreignkey="eee" value="ssss" />
                <dm-mapping:WHERE foreignkey="fff" value="tttt" />
        </dm-mapping:JOIN>
</dm-mapping:COLLECTION>
```

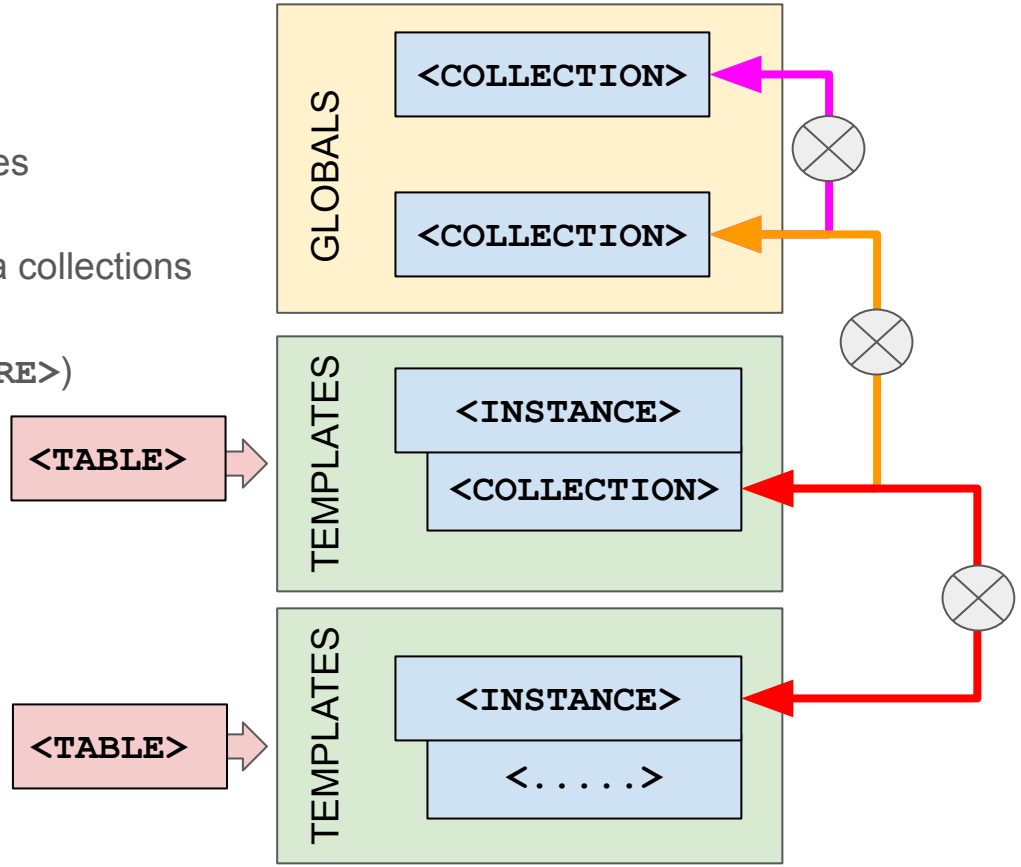


Table to Table Join

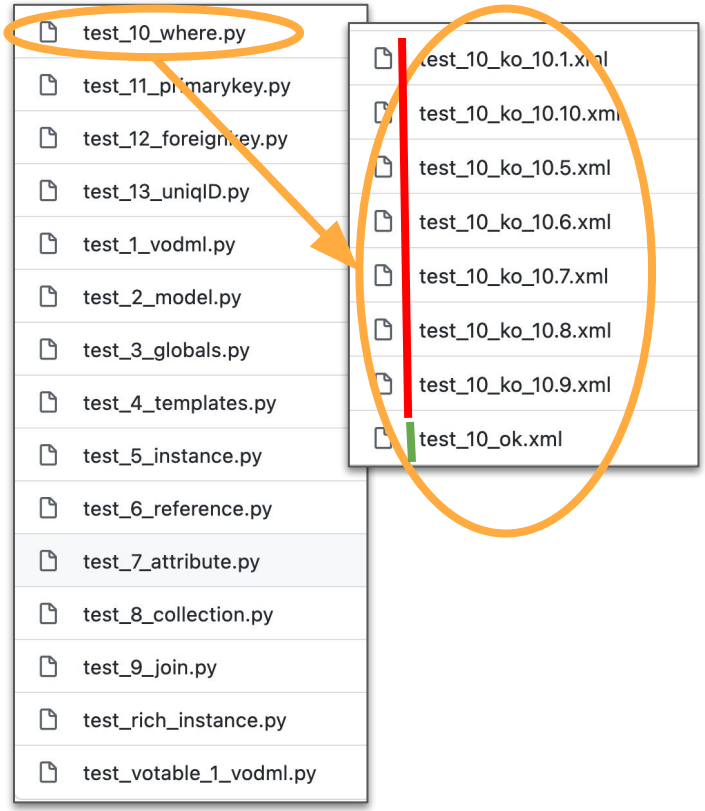
```
<dm-mapping:TEMPLATES tableref="_PKTable">
  <dm-mapping:INSTANCE dmid="_TimeSeries" dmrole="" dmtype="cube:SparseCube">
    <dm-mapping:REFERENCE dmrole="cube:DataProduct.dataset" tableref="_Datasets">
      <dm-mapping:FOREIGN_KEY ref="_pksrcid"/>
    </dm-mapping:REFERENCE>
    <dm-mapping:COLLECTION dmrole="cube:SparseCube.data">
      <dm-mapping:JOIN tableref="Results" dmref="_ts_data">
        <dm-mapping:WHERE foreignkey="_srcid" primarykey="_pksrcid" />
        <dm-mapping:WHERE foreignkey="_band" primarykey="_pkband" />
      </dm-mapping:JOIN>
    </dm-mapping:COLLECTION>
  </dm-mapping:INSTANCE>
</dm-mapping:TEMPLATES>

<dm-mapping:TEMPLATES tableref="Results">
  <dm-mapping:INSTANCE dmid="_ts_data" dmrole="" dmtype="cube:NDPoint">
    <dm-mapping:COLLECTION dmrole="cube:NDPoint.observable">
      <dm-mapping:INSTANCE dmtype="cube:Observable">
        <dm-mapping:ATTRIBUTE dmrole="cube:DataAxis.dependent" dmtype="ivoa:boolean" value="False"/>
        <dm-mapping:INSTANCE dmrole="cube:MeasurementAxis.measure" dmtype="meas:Time">
          <dm-mapping:INSTANCE dmrole="meas:Time.coord" dmtype="coords:MJD">
            <dm-mapping:ATTRIBUTE dmrole="coords:MJD.date" dmtype="ivoa:real" ref="_obstime"/>
            <dm-mapping:REFERENCE dmrole="coords:Coordinate.coordSys" dmref="_timesys"/>
```

The **`cube:SparseCube.data`** is populated with joined data

- That are instances of the table **`Results`** identified as **`dmid=_ts_data`**
- That use both columns **`_band`** and **`_srcid`** of table **`Results`** as foreign keys
- And both columns **`_pkband`** and **`_pksrcid`** of table **`PKTable`** as primary keys

Validation of the Syntax Design

- **Based on many unit tests**
  - One test suite per element
    - One snippet file with all matching patterns
    - Some failing snippets
      - Each with one pattern that must be rejected
      - Verify that tests fail for the expected reason
  - Snippets can be used as a library of standard patterns
    - Discover the syntax (reviewers will thank us)
    - Exercise the annotation
- **Document**
  - Mostly derived from schema statement
- **Validators**
  - All rules are enforced by the schema
    - No special case stated in the document (so far)
    - An XSD1.1 processor can be used as validator



XSD 1.1 in Action

```
<xs:complexType name="PrimaryKey">
  <xs:attribute type="xs:string" name="ref" />
  <xs:attribute type="xs:string" name="dmtype" />
  <xs:attribute type="xs:string" name="value" />
  <xs:assert test="./@ref  or ./@value " />
  <xs:assert test="@dmtype != '' " />
  <xs:assert test="if (./@ref) then  @ref != '' else true()" />
  <xs:assert test="(./@value and not(./@ref)) or (not(./@value) and ./@ref)" />
</xs:complexType>
```

A <**PrimaryKey**> must have

- a non empty **@dmtype**
- A **@ref** or a **@value** but not both
- **@ref** must not be empty when present

Conclusions

- **Rather Satisfied**
  - A lot of work put on since last Interop
  - Compromise built with the best of the 2 proposals
    - Simple annotations for the simple cases
    - Support all use-cases proposed in the workshop
      - Multiple joins
      - Multi filter tables

- **A lot of work still to complete**
  - Complete the specification document
  - Provide libraries connected with PyVO
  - Find name for the Standard
    - ~~Mivot, Merged Syntax~~
    - Acronym for ***Data Model Annotation Syntax for VOTables ?***
      - DMASV likely not accepted