Shanghai: From UML To VOTable

A End-to-End Trip in the VO-DML World

Santiago: Some Thoughts

Is the Mapping a *Clustering* Issue?

Mapping Targets

- **Client classes targeted by the mapping WD**
  - Simple clients: Do not care about VO-DML
  - **Advanced clients: Knows about models and parse VO-DML elements**
  - Guru client: Enable to discover models
- **Data archives targeted by the mapping WD**
  - Data delivered by Archives built on a specific model
    - CAOM
  - Annotating hierarchical data stored in a TAP service
    - Simbad, RR
  - **Identifying relevant quantities in heterogeneous datasets**
    - **GAVO, Vizier**

http://wiki.ivoa.net/twiki/bin/view/IVOA/VODML-Mapping

I'm focused on the bold-faced items

End to End Trip Outlines

1. Project Aim

a. Building a model close to a real science case
b. Getting an idea about what it does mean using VO-DML annotations.
   i. Which tool are available or missing?
   ii. Which human skill is required

2. Target

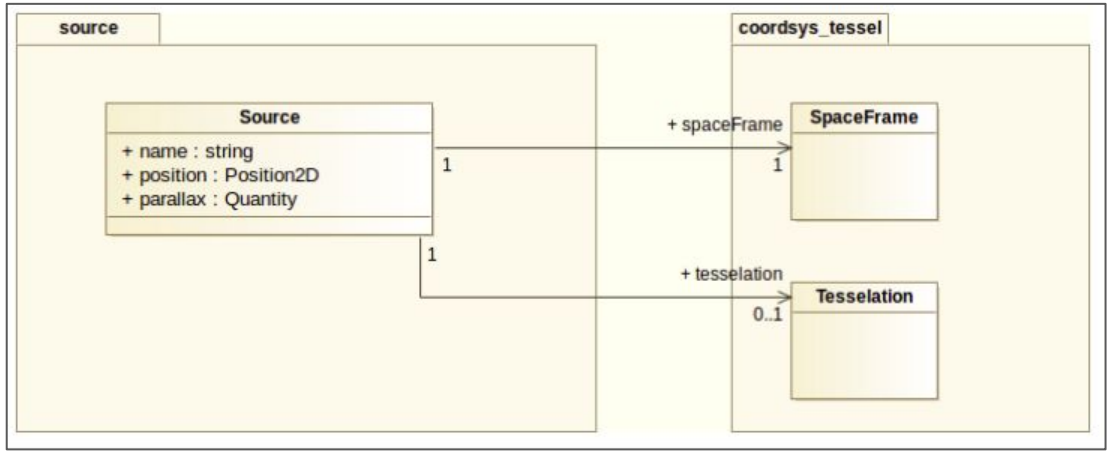
a. Replacing <COOSYS> with VO-DML annotations (Pierre's request)
b. Annotating Healpix tessellation (Markus request)

3. Procedure

a. Designing a simple model
b. Building a VO-DML/XML representation of this model
c. Annotating real data with this model

The model

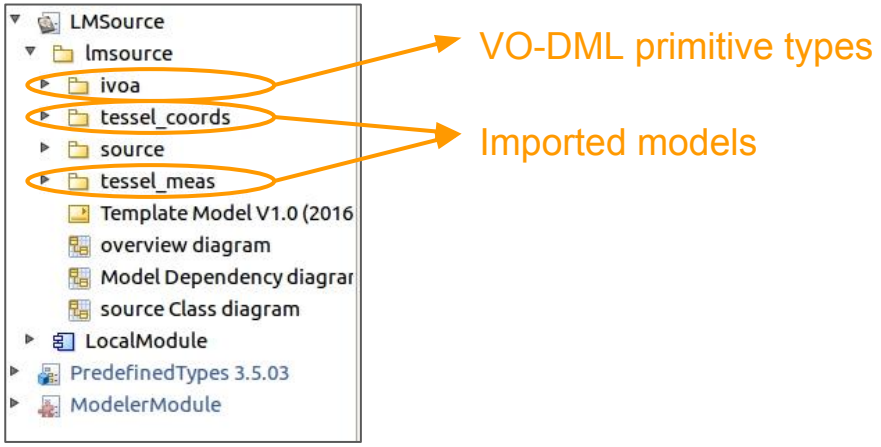
- `lmsource`: **A basic Model describing a source**
  - Name
    - also used as primary key
  - Sky coordinates
    - 2 axis: RA/DEC
    - One space frame
    - An elliptical error
  - Healpix tessellation
    - Another space frame
    - Numbering schema
    - Healpix order
    - Pixel value
  - Parallax



https://volute.g-vo.org/svn/trunk/projects/**dm/Tesselation/**

The Outcomes

- **Modeling**
  - Model designed on Modelio
  - Based on the IVOA template
  - STC imports
  - **Everything ran smoothly**

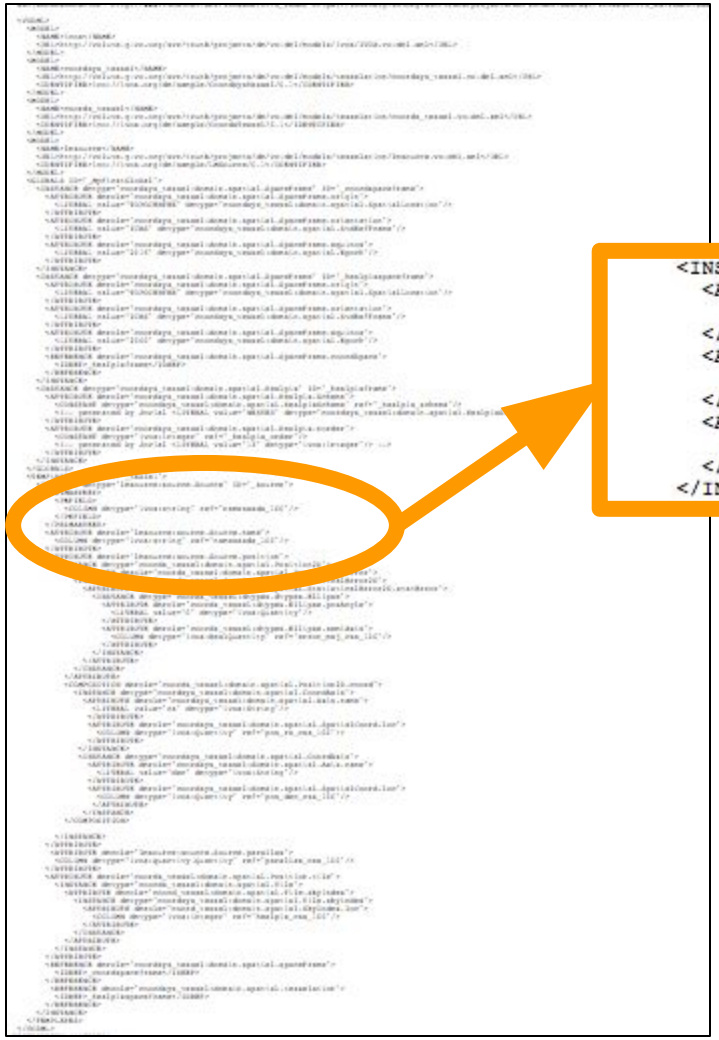


- **Serializing the model in VO-DML/XML**
  - XMI to VO-DML conversion with XSL sheets
  - VO-DML/XML is convenient way to serialize/import/export models
  - Big advantage of having a common VO schema for models
  - **Everything ran smoothly**

- **Annotating a VOTable with this model**
  - Started 6 months ago with Jovial DSL (Omar), finally acieved by hand
  - **Things ran less smoothly**
    - **!! Not tested with the last Jovail version**

What Does the Mapping Look Like

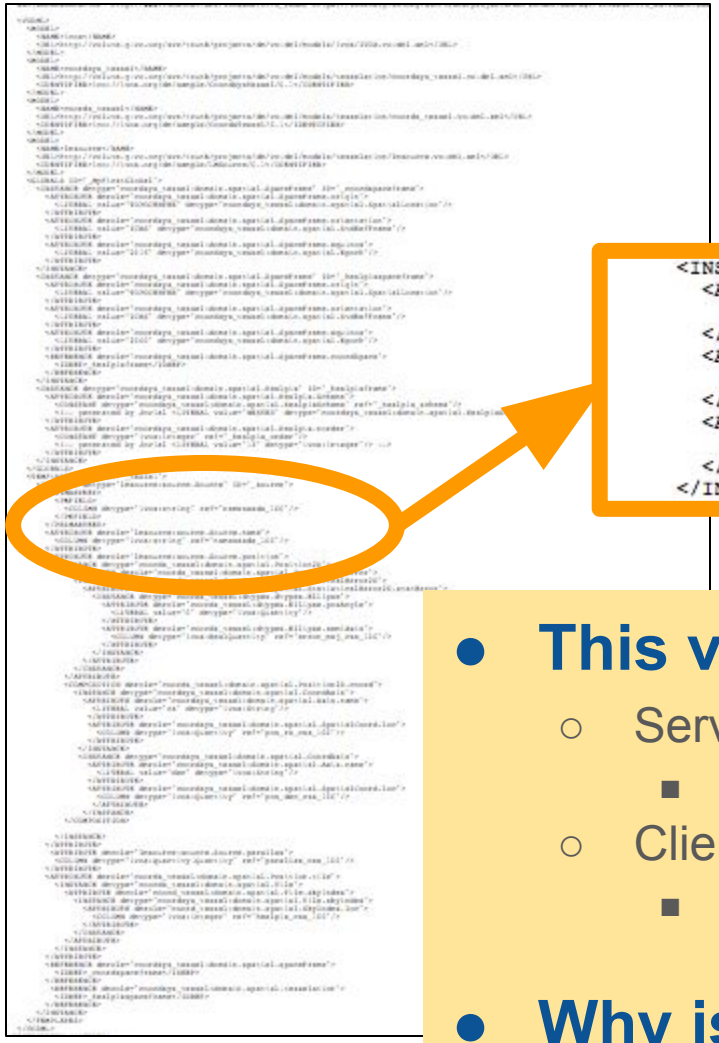
What Does the Mapping Look Like



```
<INSTANCE dmtype="coordsys_tessel:domain.spatial.SpaceFrame" ID="_coordspaceframe">
  <ATTRIBUTE dmrole="coordsys_tessel:domain.spatial.SpaceFrame.origin">
    <LITERAL value="TOPOCENTER" dmtype="coordsys_tessel:domain.spatial.SpatialLocation"/>
  </ATTRIBUTE>
  <ATTRIBUTE dmrole="coordsys_tessel:domain.spatial.SpaceFrame.orientation">
    <LITERAL value="ICRS" dmtype="coordsys_tessel:domain.spatial.StdRefFrame"/>
  </ATTRIBUTE>
  <ATTRIBUTE dmrole="coordsys_tessel:domain.spatial.SpaceFrame.equinox">
    <LITERAL value="2015" dmtype="coordsys_tessel:domain.spatial.Epoch"/>
  </ATTRIBUTE>
</INSTANCE>
```

Epoch 2015: Pierre :=)

What Does the Mapping Look Like



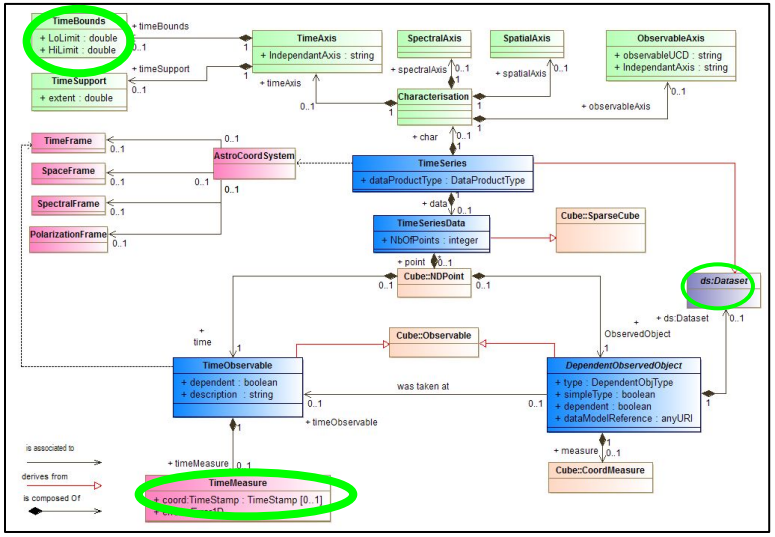
```
<INSTANCE dmtype="coordsys_tessel:domain.spatial.SpaceFrame" ID="_coordspaceframe">
  <ATTRIBUTE dmrole="coordsys_tessel:domain.spatial.SpaceFrame.origin">
    <LITERAL value="TOPOCENTER" dmtype="coordsys_tessel:domain.spatial.SpatialLocation"/>
  </ATTRIBUTE>
  <ATTRIBUTE dmrole="coordsys_tessel:domain.spatial.SpaceFrame.orientation">
    <LITERAL value="ICRS" dmtype="coordsys_tessel:domain.spatial.StdRefFrame"/>
  </ATTRIBUTE>
  <ATTRIBUTE dmrole="coordsys_tessel:domain.spatial.SpaceFrame.equinox">
    <LITERAL value="2015" dmtype="coordsys_tessel:domain.spatial.Epoch"/>
  </ATTRIBUTE>
</INSTANCE>
```

Epoch 2015: Pierre :=)

- **This verbosity frightens some of us.**
  - Service providers
    - How to generate such a thing for all of my datasets?
  - Client developers
    - What cost for parsing this?
- **Why is it so verbose?**

Building the VO-DML mapping block

MODEL



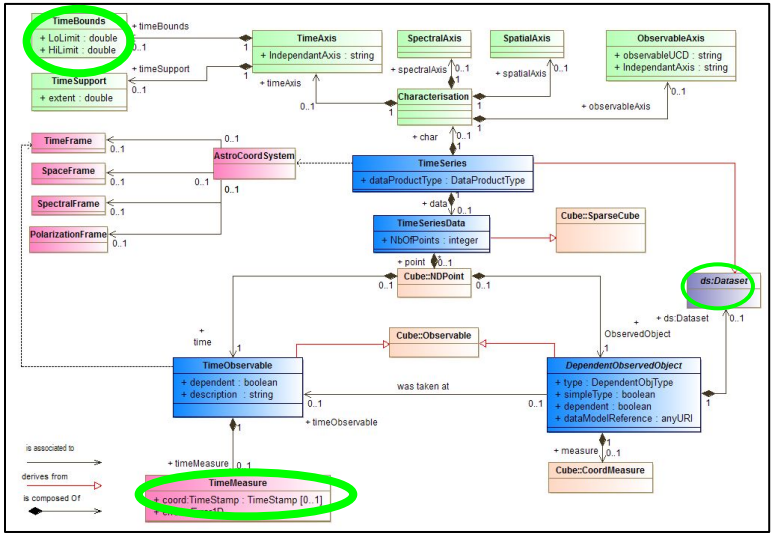
VOTABLE

<Table #1>

<wable #2>

Building the VO-DML mapping block

MODEL



VOTABLE

<VODML>

<MODELS>

List of used models

<GLOBALS>

Instances valid everywhere

<TEMPLATES table#1>

Templates of objects contained in table #1

<TEMPLATES table#2>

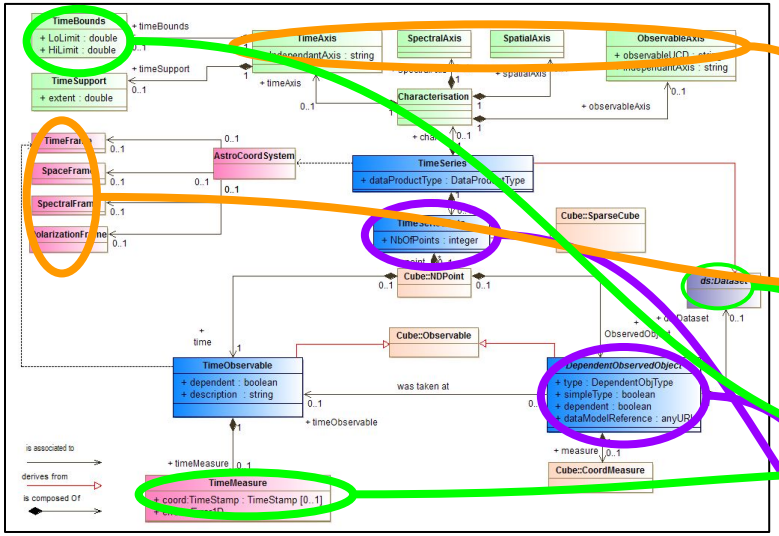
Templates of objects contained in table #2

<Table #1>

<wable #2>

Building the VO-DML mapping block

MODEL



VOTABLE

<VODML>

<MODELS>

List of used models

<GLOBALS>

Instances valid everywhere

<TEMPLATES table#1>

Templates of objects contained in table #1

<TEMPLATES table#2>

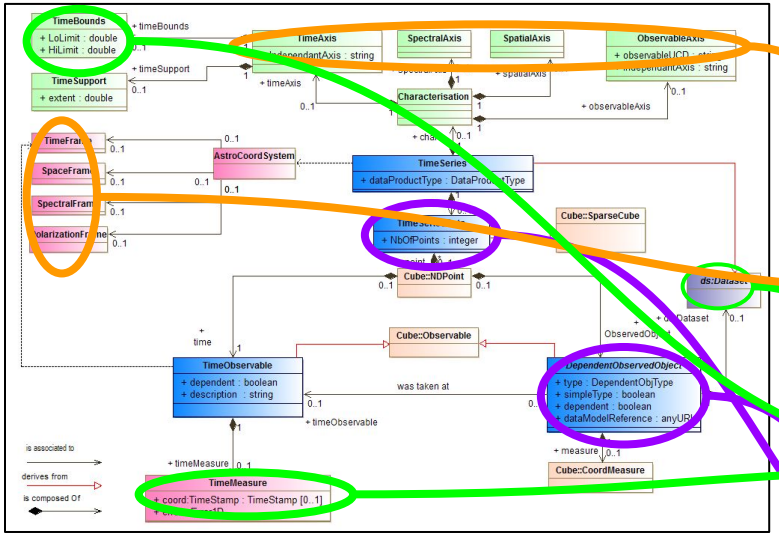
Templates of objects contained in table #2

<Table #1>

<wable #2>

Building the VO-DML mapping block

MODEL



VOTABLE

<VODML>

<MODELS>

List of used models

<GLOBALS>

Instances valid everywhere

<TEMPLATES table#1>

Templates of objects contained in table #1

<TEMPLATES table#2>

Templates of objects contained in table #2

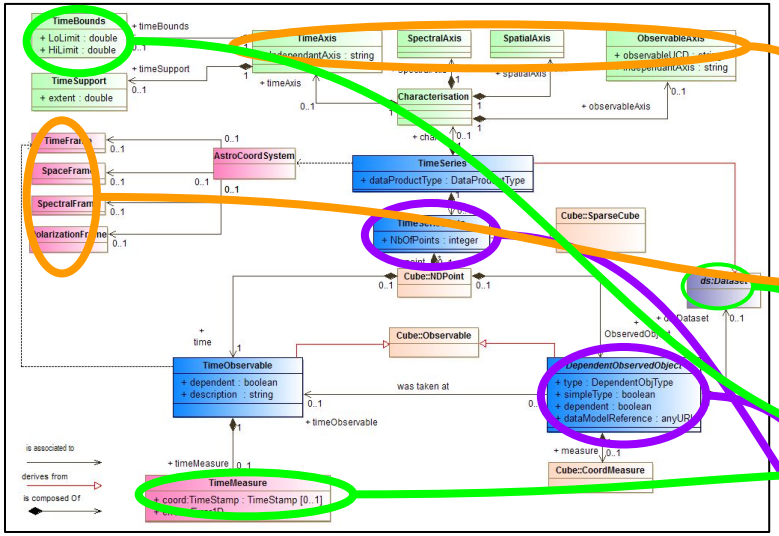
<Table #1>

<wable #2>

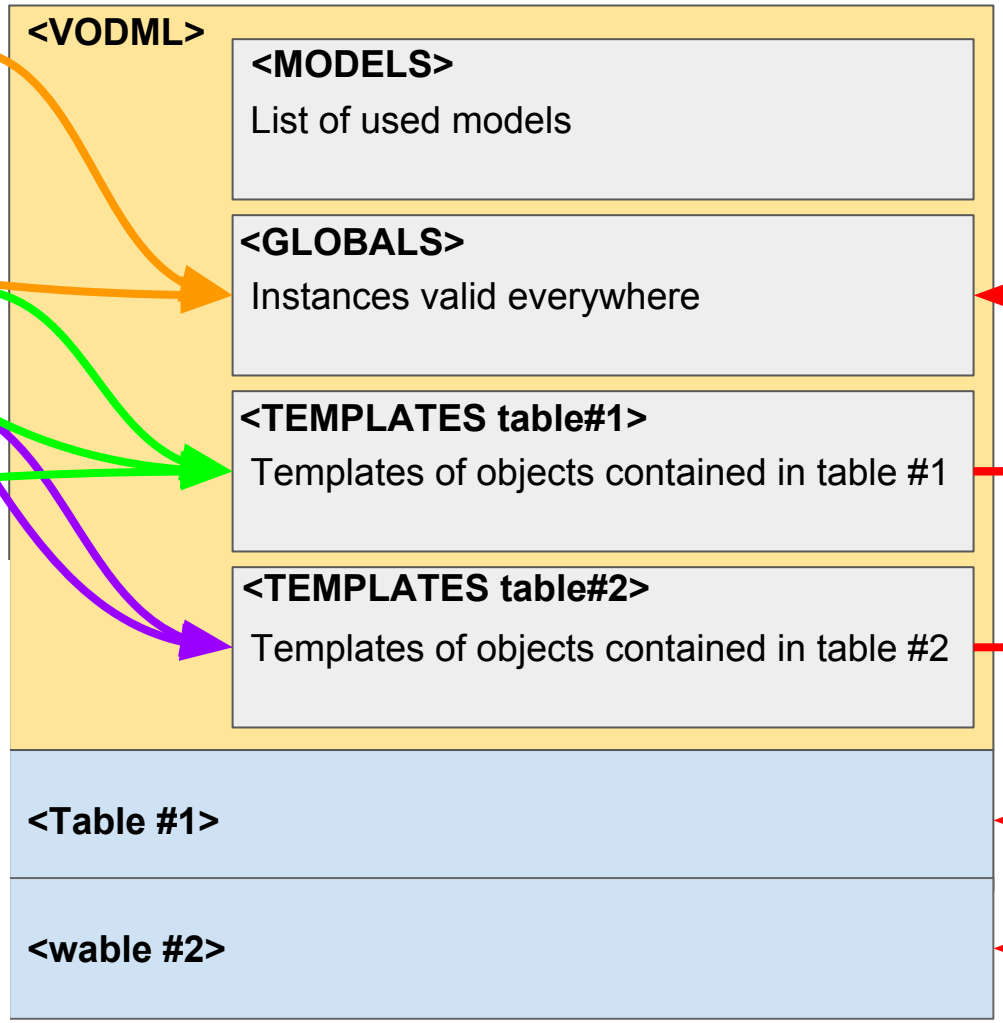
Data pointers

Building the VO-DML mapping block

MODEL



VOTABLE

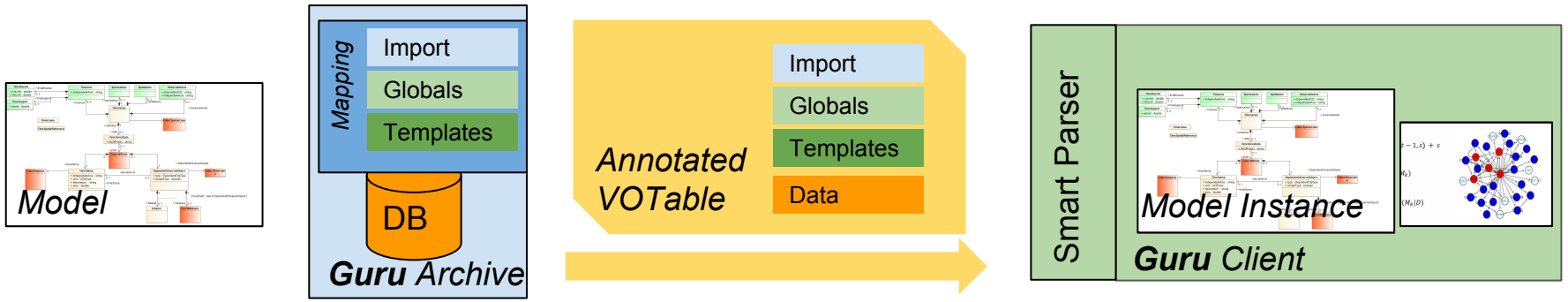


- **Model components are reported in mapping blocks**
  - Difficult to automate
- **Data pointers are included in model leaves**
  - Different pointers for `FIELD/PARAM/GLOBALS/LITERAL`
  - Keys for joining data

The high verbosity is an unavoidable consequence of this mechanism.

Mapping: Approaches and Expectations

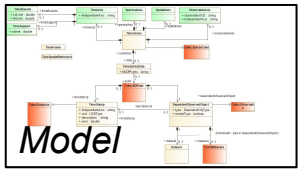
Guru Usage (Object Relational Mapping)

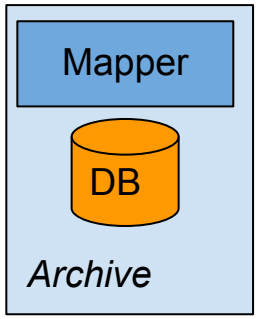


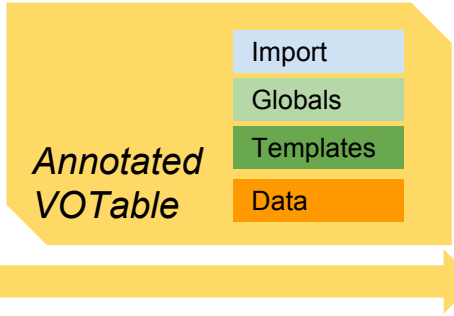
The whole model semantics must be carried with the VOTable:
It can't be anything but chatty.

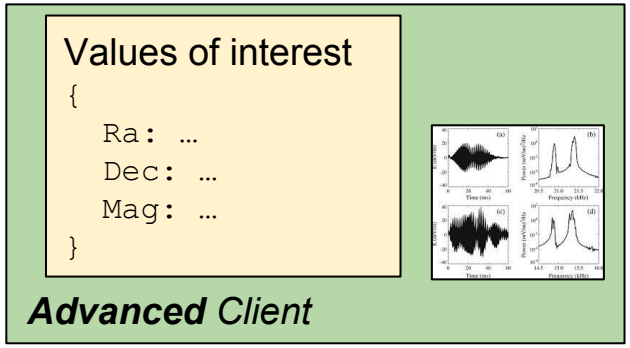
Mapping: Approaches and Expectations

Advanced Usage





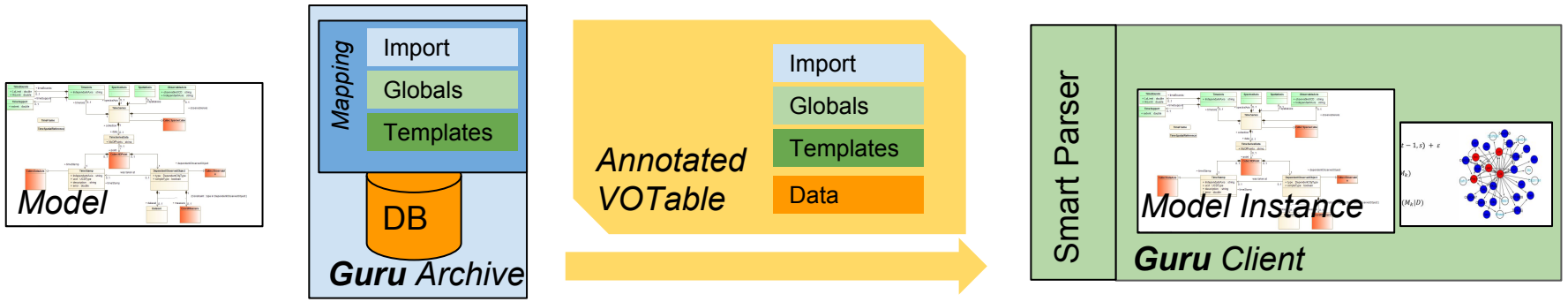




Do not need to carry all OO modeling subtleties
Just need populate some model leaves with values

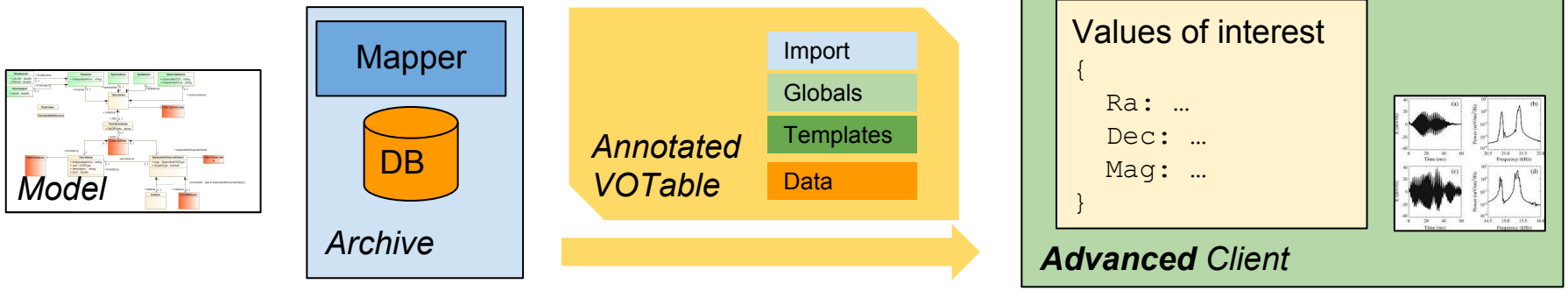
Mapping: Approaches and Expectations

Guru Usage (Object Relational Mapping)



The whole model semantics must be carried with the VOTable:
It can't be anything but chatty.

Advanced Usage



Do not need to carry all OO modeling subtleties
Just need populate some model leaves with values

Conclusions

Annotation = model serialization adapted to the actual data

- **My evaluation**
  - Strength
    - Grouping annotations in one block facilitates the job for all clients (simple, adv. and guru).
    - Annotated VOTables are consistent products which can be processed independently of any other resource
    - Successfully tested on different (and complex) use cases

Conclusions

Annotation = model serialization adapted to the actual data

- **My evaluation**
  - Strength
    - Grouping annotations in one block facilitates the job for all clients (simple, adv. and guru).
    - Annotated VOTables are consistent products which can be processed independently of any other resource
    - Successfully tested on different (and complex) use cases
  - Flaws
    - Need to validate the mapping against the model
    - Annotations are quite complex since all modeling features are reported in it
    - Difficult to write and test
    - Painfull to apply to existing archives

Conclusions

Annotation = model serialization adapted to the actual data

- **My evaluation**
  - Strength
    - Grouping annotations in one block facilitates the job for all clients (simple, adv. and guru).
    - Annotated VOTables are consistent products which can be processed independently of any other resource
    - Successfully tested on different (and complex) use cases
  - Flaws
    - Need to validate the mapping against the model
    - Annotations are quite complex since all modeling features are reported in it
    - Difficult to write and test
    - Painfull to apply to existing archives

- **We must decide on the endorsement of this proposal**
  - Accepting the current approach even oversized for many concrete use cases
  - Reducing the scope of the current approach to simplify the syntax
  - Drawing a middle path

No Consensus on those Approaches

- **Never ended discussion**
  -  **We** could annotate our data with something simpler
  -  **Yes**, **but** you couldn't build a model instance with something simpler
  -  **Yes**, **but** we could annotate our data with something simpler
  -  **Yes but**, you …. `while(true == true){}`
- **What are the possible outcomes?**
  - Accepting the ORM approach even its is oversized for most of the current use cases
  - Reducing the scope of the ORM approach
  - Drawing a middle path

Using the VOTable Schema Import

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
    xmlns="http://www.ivoa.net/xml/VOTable/v1.4_vodml" targetNamespace="http://www.ivoa.net/xml/VOTable/v1.4_vodml">
    <!--
        Include the VOML mapping within a unique namespace (workaround for validator issues) (LM)
     -->
    <xs:include schemaLocation="VODML-mapping.xsd"/>
```

The mapping schema is imported by VOTable 1.4 schema

We could have a lite mapping schema in addition to the actual one
The difference between both would the include statement in the VOTable schema

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
    xmlns="http://www.ivoa.net/xml/VOTable/v1.4_vodml" targetNamespace="http://www.ivoa.net/xml/VOTable/v1.4_vodml">
    <!--
        Include the VOML mapping within a unique namespace (workaround for validator issues) (LM)
     -->
    <xs:include schemaLocation="VODML-ORM-mapping.xsd"/>
```

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
    xmlns="http://www.ivoa.net/xml/VOTable/v1.4_vodml" targetNamespace="http://www.ivoa.net/xml/VOTable/v1.4_vodml">
    <!--
        Include the VOML mapping within a unique namespace (workaround for validator issues) (LM)
     -->
    <xs:include schemaLocation="VODML-LITE-mapping.xsd"/>
```