VODML Mapping Update

Mark Cresitello-Dittmar, SAO
Tom Donaldson, STScI
Omar Laurino, SAO
Gerard Lemson, JHU
Laurent Michel, OAS

New Mapping Syntax

- **Reminder: Mapping syntax defines how to annotate a VOTable to serialize objects of a VODML model.**

- **Previous proposed syntax minimized the number of new elements added to the VOTable spec.**
    - Low impact on the xsd, so it was easy to parse, but…
    - Very complex to document and code to because understanding the meaning of the content was very context dependent.

- **New syntax much more explicit**
    - Adds several new elements that map directly to VODML concepts.
    - Aims to remove context-dependent logic
    - Number of new elements makes it clear we're solving a complex problem
    - Impact on parsers and data providers still limited due to isolating all new elements in a single section of the VOTable.

New Developments

- **New Mapping Document**
  - Includes continuous integration paradigm and is managed in GitHub
- **Multiple people have created annotated VOTables for multiple valid VODML models**
  - Several examples embedded in mapping document.
  - Catch-all for most constructs (GL)
  - Cube model(MCD)
  - Tessellation (LM)
- **Annotation automation tools (OL)**
  - Helped to create many of the above VOTables
- **Parser prototype (TD)**
  - Web service and client for parsing and displaying annotated VOTables
  - Handles the above examples (except the FITS data)
  - Results promising. More work needed to explore all use cases.

New Developments (2)

- **At least three independent interoperable implementations**
  - Model
  - Create format-independent instances
  - Annotate VOTable
  - Parse VOTable and faithfully reconstruct instances
  - Schema validation
- **Validation is currently the weakest link**
  - You can validate against XSD with standard XML tools
  - No VO-DML specific validation of instances just yet
- **VOTable and VO-DML Mapping schemata**
  - Separated into individual schemata
  - Transparent to clients (only one namespace, as usual)
  - More sustainable development
  - Need to decide whether to “import” or “include”

Mapping Document Reboot

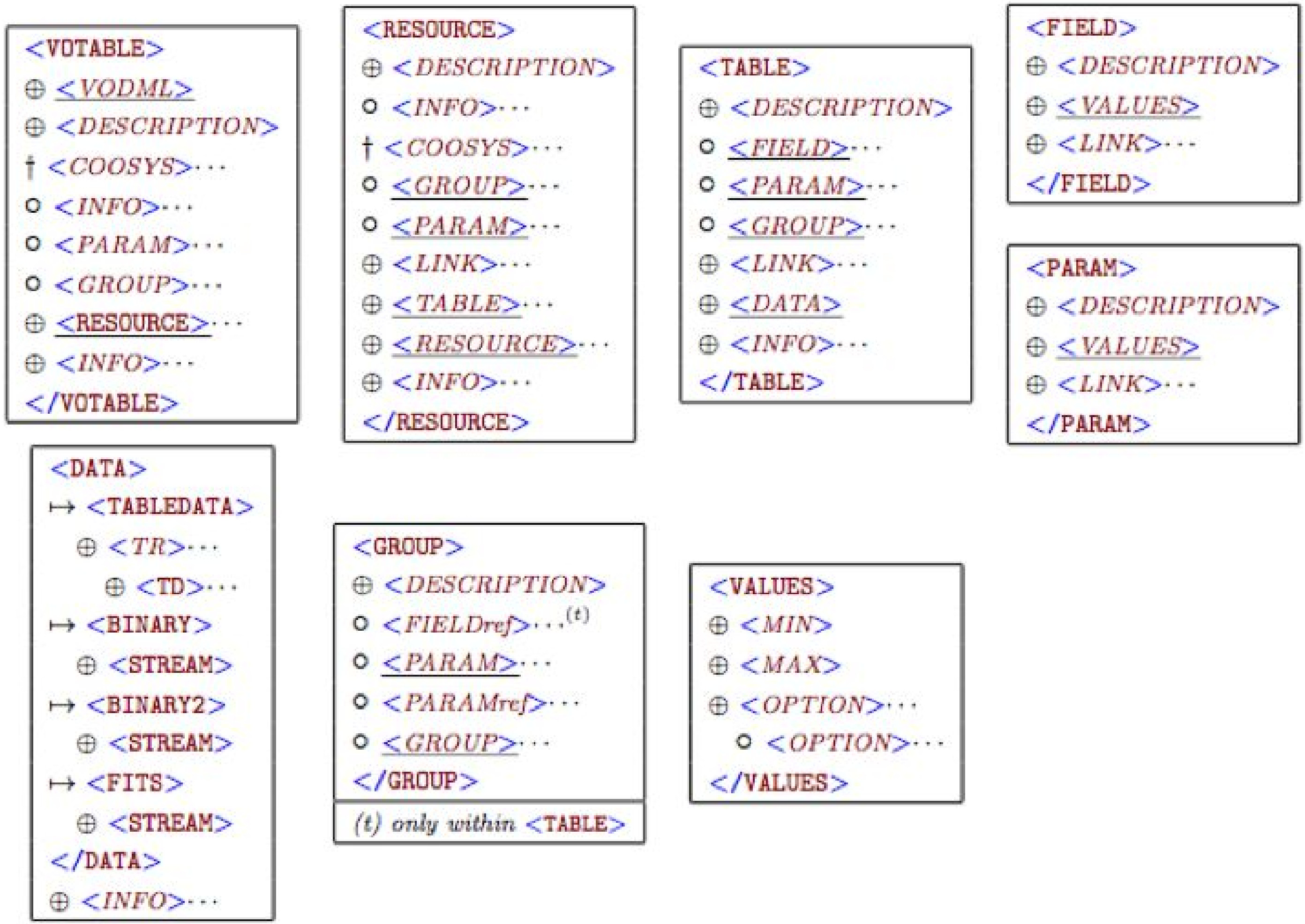
- **Rewrote from scratch with the new syntax**
- **Distributed Version Control of the source files (git/GitHub)**
- **MarkDown source with HTML and PDF outputs (like ivoatex, but w/ MD)**
- **Special integration with VO-DML:**
  - write examples in DSL
  - produce annotation during compilation
  - Include external files
- **Continuous Integration with ad-hoc cloud web application:**
  - Build HTML document upon push
  - Link to Diff
  - Keep history of changes

Mapping Document Reboot (2)

- **Extension and consolidation of use cases: many concrete and abstract use cases were added or rewritten**
- **Section 3 (The need for a mapping language) mostly unchanged**
- **Two-way mapping description:**
  - Detailed description of new XML elements
  - How to map VO-DML types to XML elements
- **Examples:**
  - Simple example with minimal annotations
  - Complex example with advanced Object Relational Mapping patterns, multiple tables

VOTable Syntax Summary

- **VOTable spec summarizes syntax in section 7**



```
<VOTABLE>
⊕ <VODML>
⊕ <DESCRIPTION>
† <COOSYS>···
○ <INFO>···
○ <PARAM>···
○ <GROUP>···
⊕ <RESOURCE>···
⊕ <INFO>···
</VOTABLE>
```

```
<RESOURCE>
⊕ <DESCRIPTION>
○ <INFO>···
† <COOSYS>···
○ <GROUP>···
○ <PARAM>···
⊕ <LINK>···
⊕ <TABLE>···
⊕ <RESOURCE>···
⊕ <INFO>···
</RESOURCE>
```

```
<TABLE>
⊕ <DESCRIPTION>
○ <FIELD>···
○ <PARAM>···
○ <GROUP>···
⊕ <LINK>···
⊕ <DATA>
⊕ <INFO>···
</TABLE>
```

```
<FIELD>
⊕ <DESCRIPTION>
⊕ <VALUES>
⊕ <LINK>···
</FIELD>
```

```
<PARAM>
⊕ <DESCRIPTION>
⊕ <VALUES>
⊕ <LINK>···
</PARAM>
```

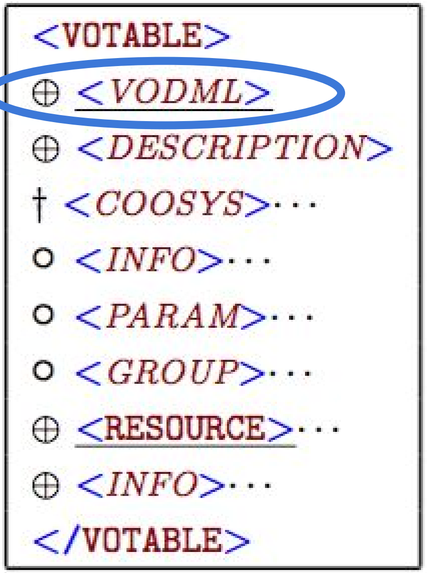
```
<DATA>
↦ <TABLEDATA>
    ⊕ <TR>···
        ⊕ <TD>···
↦ <BINARY>
    ⊕ <STREAM>
↦ <BINARY2>
    ⊕ <STREAM>
↦ <FITS>
    ⊕ <STREAM>
</DATA>
⊕ <INFO>···
```

```
<GROUP>
⊕ <DESCRIPTION>
○ <FIELDref>···(t)
○ <PARAM>···
○ <PARAMref>···
○ <GROUP>···
</GROUP>
(t) only within <TABLE>
```

```
<VALUES>
⊕ <MIN>
⊕ <MAX>
⊕ <OPTION>···
    ○ <OPTION>···
</VALUES>
```

VODML in VOTable Syntax

- **All new elements are under a single VODML element at the beginning of the VOTable.**
    - Other than the top level VODML element, the new elements can be defined in a separate xsd that gets included into the VOTable xsd.



<VOTABLE>
⊕ <VODML>
⊕ <DESCRIPTION>
† <COOSYS>···
○ <INFO>···
○ <PARAM>···
○ <GROUP>···
⊕ <RESOURCE>···
⊕ <INFO>···
</VOTABLE>

- **VODML elements have all new names (no reuse from VOTable).**

- **VODML hierarchy draft at** https://volute.g-vo.org/svn/trunk/projects/dm/vo-dml/doc/VO-DML_mapping_syntax_summary.pdf

Syntax - New Elements

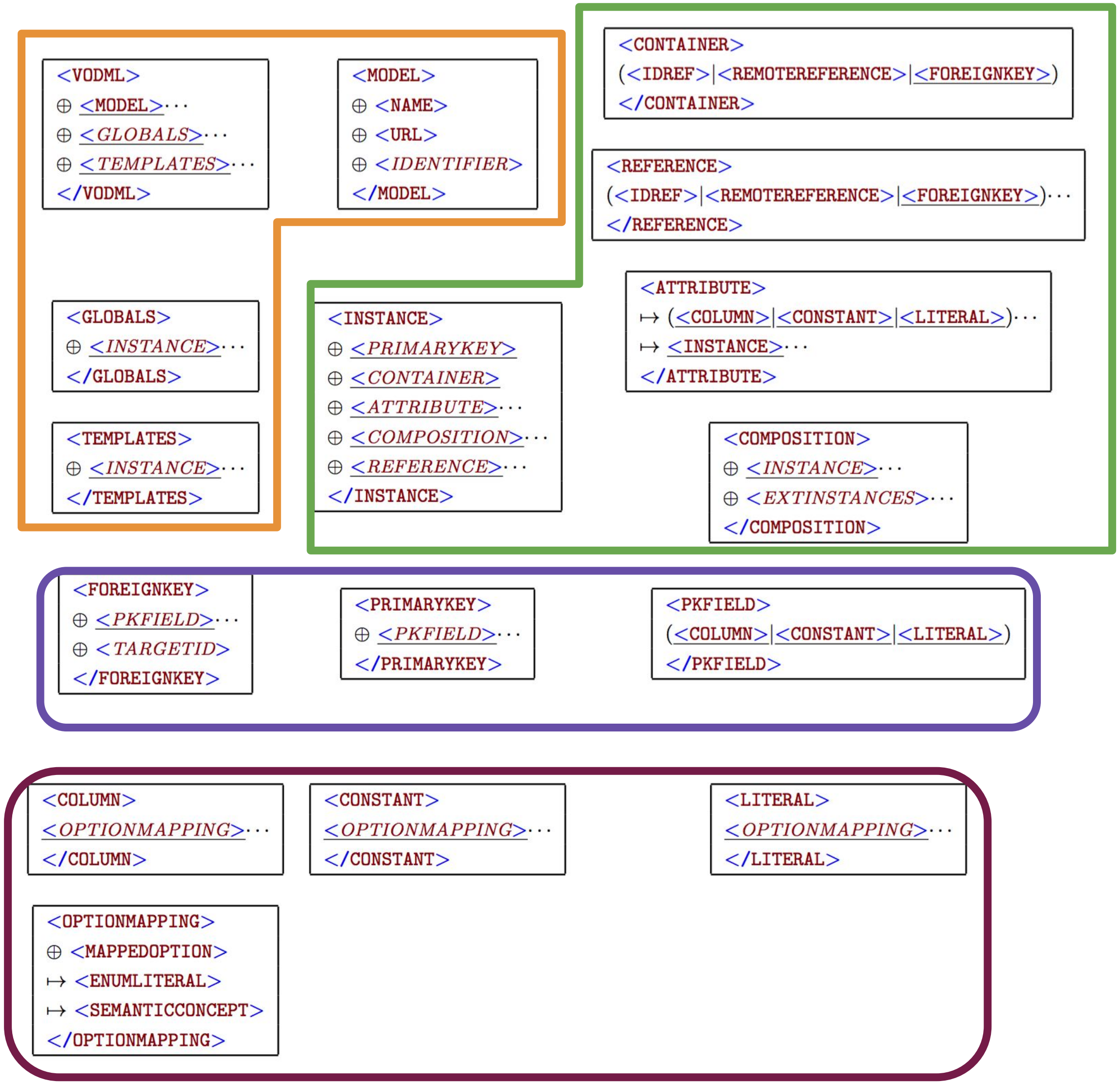
- **Structure explicitly matches VODML model concepts.**

Core elements

Instance content

Foreign key refs

Primitive values



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

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

```
<GLOBALS>
⊕ <INSTANCE>···
</GLOBALS>
```

```
<TEMPLATES>
⊕ <INSTANCE>···
</TEMPLATES>
```

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

```
<REFERENCE>
(<IDREF>|<REMOTEREFERENCE>|<FOREIGNKEY>)···
</REFERENCE>
```

```
<INSTANCE>
⊕ <PRIMARYKEY>
⊕ <CONTAINER>
⊕ <ATTRIBUTE>···
⊕ <COMPOSITION>···
⊕ <REFERENCE>···
</INSTANCE>
```

```
<ATTRIBUTE>
↦ (<COLUMN>|<CONSTANT>|<LITERAL>)···
↦ <INSTANCE>···
</ATTRIBUTE>
```

```
<COMPOSITION>
⊕ <INSTANCE>···
⊕ <EXTINSTANCES>···
</COMPOSITION>
```

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

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

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

```
<COLUMN>
<OPTIONMAPPING>···
</COLUMN>
```

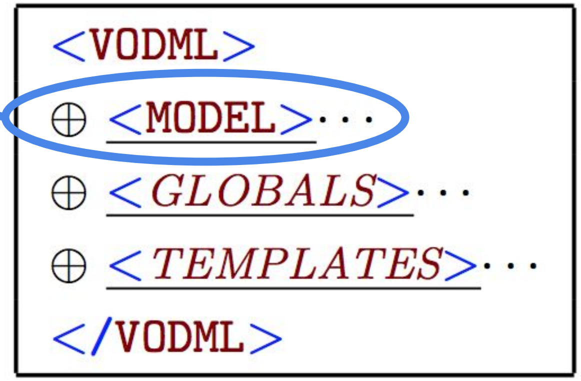
```
<CONSTANT>
<OPTIONMAPPING>···
</CONSTANT>
```

```
<LITERAL>
<OPTIONMAPPING>···
</LITERAL>
```

```
<OPTIONMAPPING>
⊕ <MAPPEDOPTION>
↦ <ENUMLITERAL>
↦ <SEMANTICCONCEPT>
</OPTIONMAPPING>
```

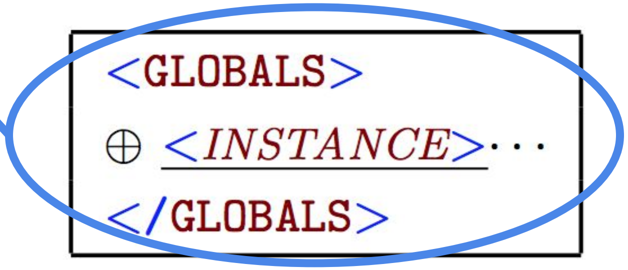
Syntax - Core Elements

- **Pointers to model definitions**



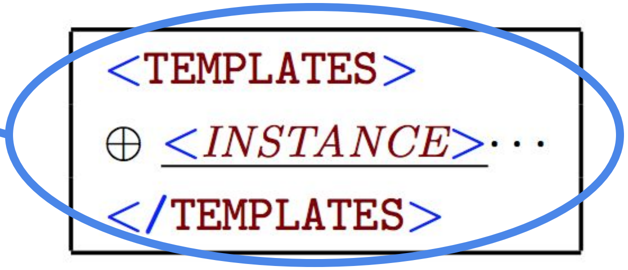
```
<VODML>
⊕ <MODEL>···
⊕ <GLOBALS>···
⊕ <TEMPLATES>···
</VODML>
```

- **Model instances that do not depend on TABLE data**
  - Complete VODML instances can be conveyed here without any actual TABLEs.



```
<GLOBALS>
⊕ <INSTANCE>···
</GLOBALS>
```

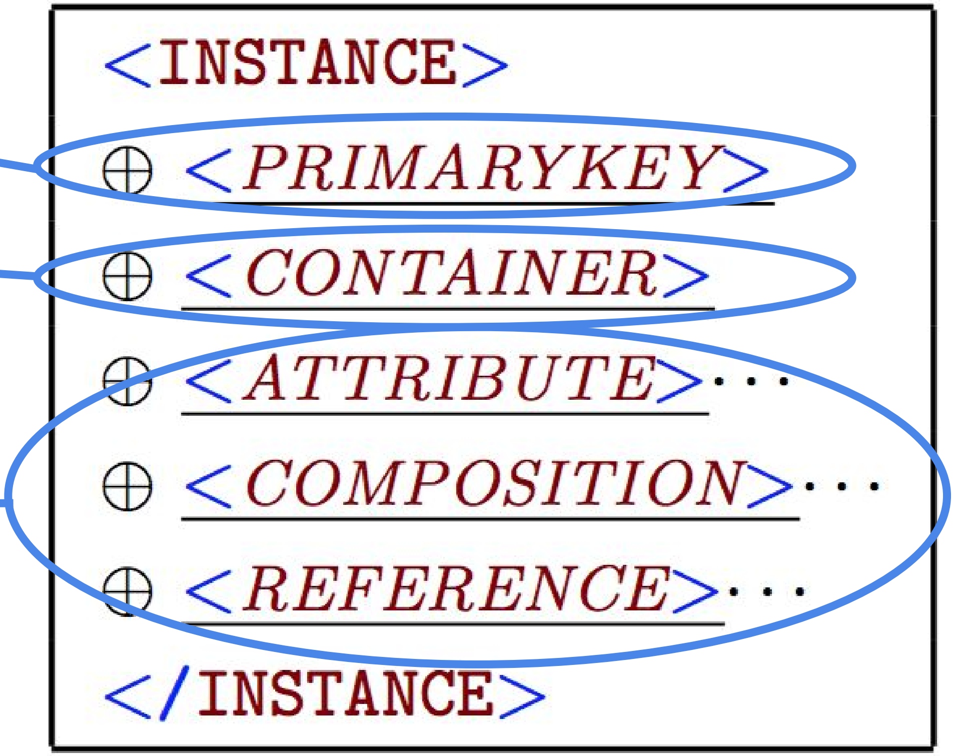
- **Instance templates created once per row using data from the row**



```
<TEMPLATES>
⊕ <INSTANCE>···
</TEMPLATES>
```

Syntax - Instance Definition

- Used for foreign key reference
- Reference to parent
- 3 ways to specify the attributes of the instance



```
<INSTANCE>
⊕ <PRIMARYKEY>
⊕ <CONTAINER>
⊕ <ATTRIBUTE>···
⊕ <COMPOSITION>···
⊕ <REFERENCE>···
</INSTANCE>
```

Annotation Experiences - Omar

Jovial:

- **Java/Groovy application and library**
- **Reference implementation for VO-DML and Mapping Document**
- **Domain Specific Language for Models and Instances**
- **Support Data Providers:**
  - Script annotations
  - Abstract from complexity of standards and syntax
- **Shield examples in mapping document from changes to syntax**
- **Target different serialization formats (VOTable, XML, JSON, YAML)**
- **Quick turnaround for end-to-end modeling efforts:**
  - Successfully applied to STC2, Cube, and TimeSeries
- **Will address more use cases as we approach PR**
  - Model-specific DSLs
  - Annotation “bootstrap”

Annotation Experiences - Omar (2)

- **End-to-end modeling of complex domains has become easier with specialized tools**
  - model (PH’s VODSL, OL’s Jovial, UML modeling)
  - instantiate (OL’s Jovial)
  - map/serialize (OL’s Jovial)
  - test against requirements and for usability/clarity
  - go back to model
- **Enables patterns like Test Driven Modeling or Rapid Model Development**

Parser Experiment - Goals

VODML (and mapping it to VOTable) is complex.

- Assess the complexity of parsing and understanding the semantics of the new syntax.
- Provide alternate view(s) for human readability.
- Show that a client can understand mapped model instances from multiple providers (interoperability)
- Identify validations beyond those enforced by schema.

Parser - Design/Plan

(Complete)

- **Adapt the MAST Portal VOTable parser**
  - Server-side C#
  - Translate annotations to JSON, but **do not** resolve references.
- **Implement Javascript client that can display instances**
  - Client will resolve references and expand templates for each data row.
  - Display multi-table VOTables.

(In Progress)

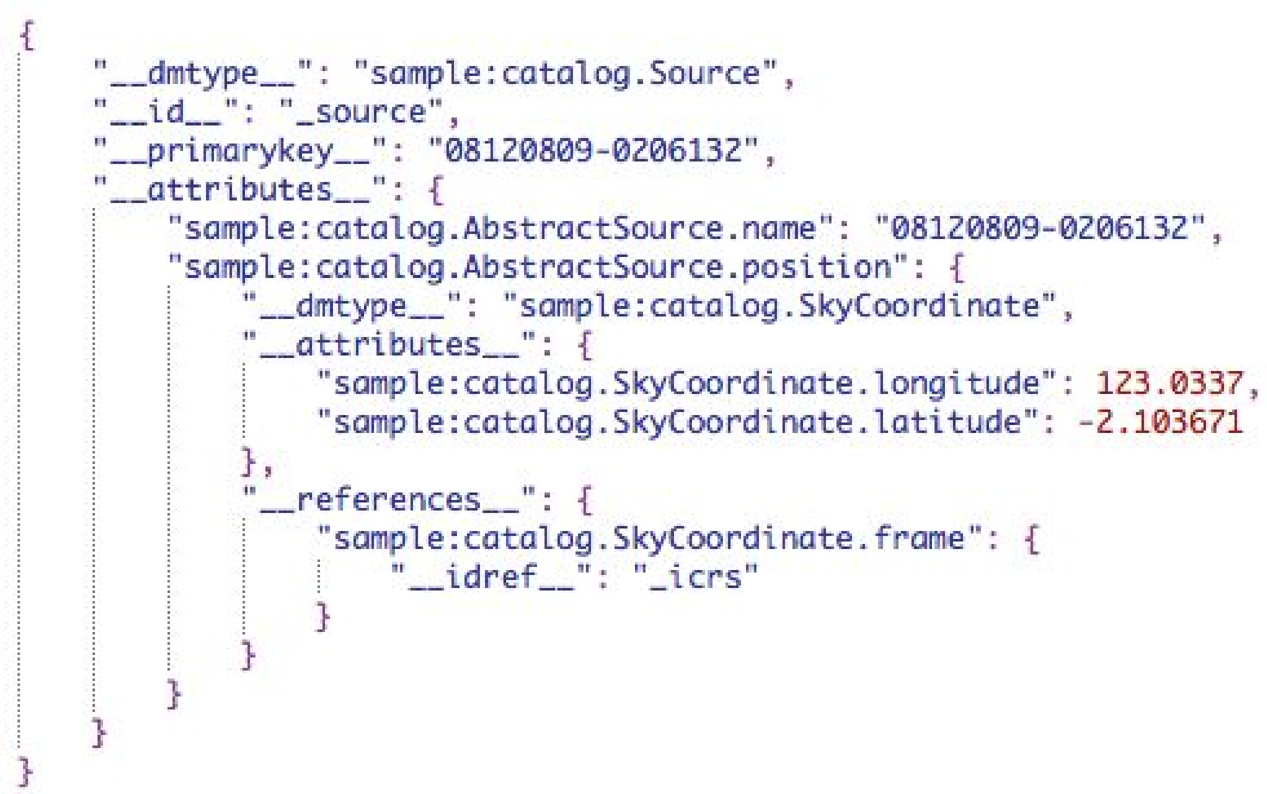
- **Handle foreign key references.**
- **Model-aware displays and validation**
  - End-to-end application of “real world” examples with models like Source and CAOM (subject to the complexities of defining the actual models).
- **Full server-side validation and dereferencing**

Parser - Server produces JSON

Server and client instructions at:

https://volute.g-vo.org/svn/trunk/projects/dm/vo-dml/mapping/MAST%20VODML%20Parser/VODML%20Parser.html

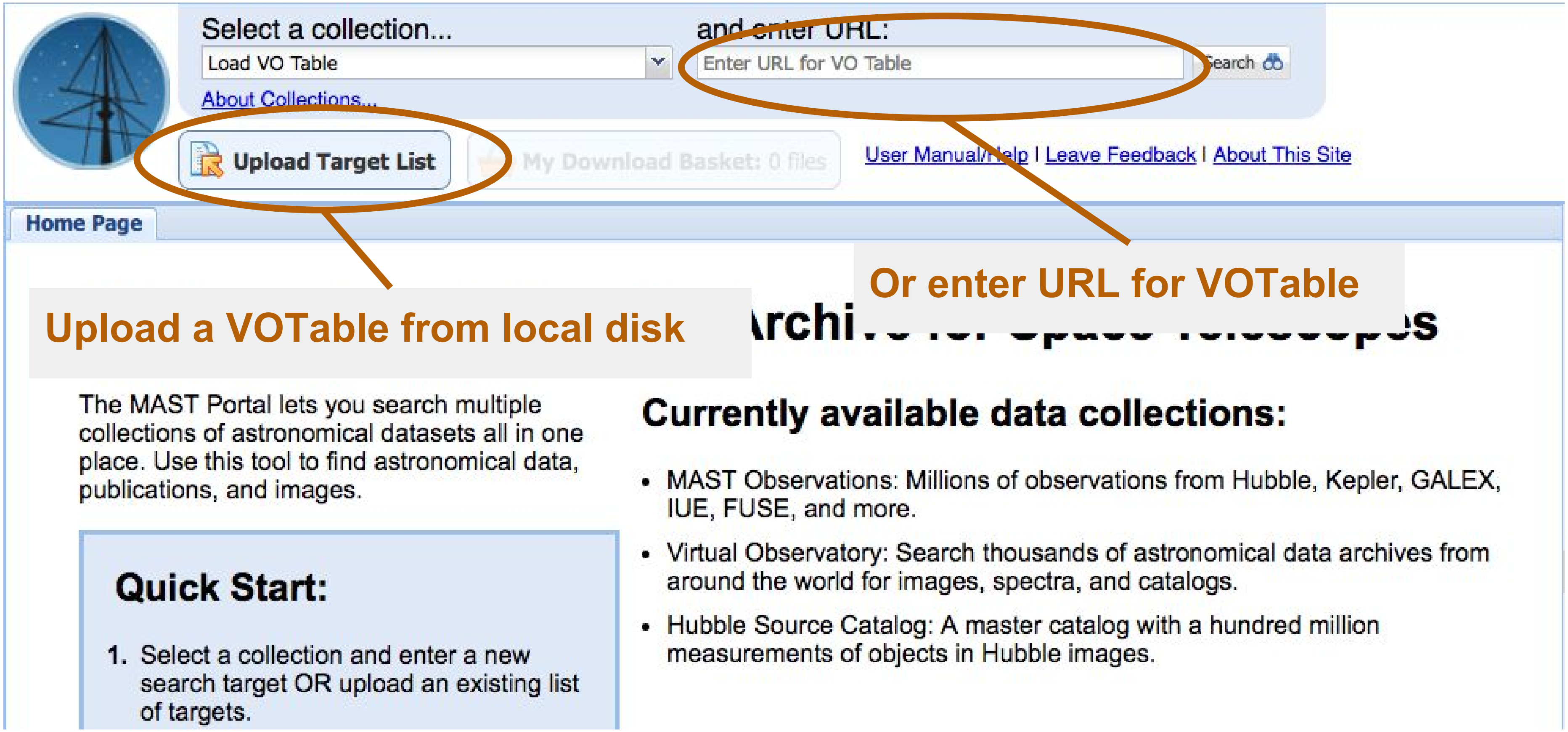
JSON for Source model instance:



```
{
    "__dmtype__": "sample:catalog.Source",
    "__id__": "_source",
    "__primarykey__": "08120809-0206132",
    "__attributes__": {
        "sample:catalog.AbstractSource.name": "08120809-0206132",
        "sample:catalog.AbstractSource.position": {
            "__dmtype__": "sample:catalog.SkyCoordinate",
            "__attributes__": {
                "sample:catalog.SkyCoordinate.longitude": 123.0337,
                "sample:catalog.SkyCoordinate.latitude": -2.103671
            },
            "__references__": {
                "sample:catalog.SkyCoordinate.frame": {
                    "__idref__": "_icrs"
                }
            }
        }
    }
}
```

Parser - Client Based on MAST Portal

https://masttest.stsci.edu/vodml/Mashup/Clients/Mast/Portal.html



Upload a VOTable from local disk

Or enter URL for VOTable

The MAST Portal lets you search multiple collections of astronomical datasets all in one place. Use this tool to find astronomical data, publications, and images.

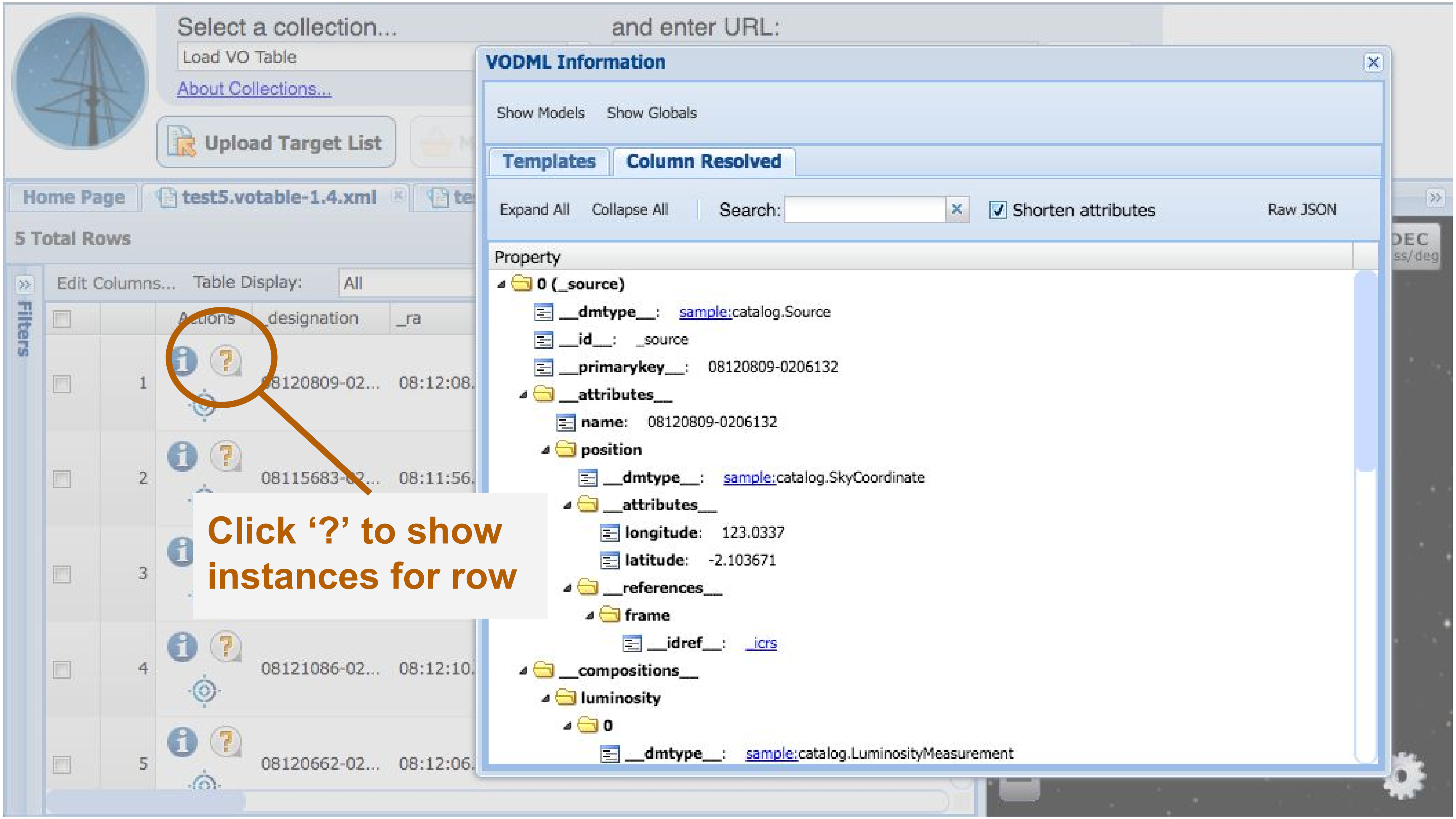
Quick Start:

1. Select a collection and enter a new search target OR upload an existing list of targets.

Currently available data collections:

- MAST Observations: Millions of observations from Hubble, Kepler, GALEX, IUE, FUSE, and more.
- Virtual Observatory: Search thousands of astronomical data archives from around the world for images, spectra, and catalogs.
- Hubble Source Catalog: A master catalog with a hundred million measurements of objects in Hubble images.

Parser - Client (cont’d)



5 Total Rows

□		Actions	designation	_ra
□	1		08120809-02...	08:12:08.
□	2		08115683-02...	08:11:56.
□	3			
□	4		08121086-02...	08:12:10.
□	5		08120662-02...	08:12:06.

```
0 (_source)
    __dmtype__:  sample:catalog.Source
    __id__:  _source
    __primarykey__:  08120809-0206132
    __attributes__
        name:  08120809-0206132
        position
            __dmtype__:  sample:catalog.SkyCoordinate
            __attributes__
                longitude:  123.0337
                latitude:  -2.103671
            __references__
                frame
                    __idref__:  _icrs
    __compositions__
        luminosity
            0
                __dmtype__:  sample:catalog.LuminosityMeasurement
```

Click ‘?’ to show instances for row

Parser - Results So Far

New syntax is significant improvement over old.

- **Old syntax was easy to parse, but hard to understand.**
- **New syntax is tedious (but not hard) to parse**
    - Tedium is due to number of new elements.
    - Some implementations may be easier than others.
- **New syntax is much easier to understand**
    - Due to direct, explicit, mapping from new elements to VODML concepts.
    - There is still complexity.
    - Learning curve is for all aspects of VODML, not just the mapping.
    - Basic cases are not too complex.
        - E.g., substitution of FIELD values in templates is straightforward and delineates objects more clearly than UTypes.

Parser - Results So Far (2)

- **Some simplifications are possible if we decide certain features or explicit structures are not needed.**
  - E.g., ATTRIBUTE could be used instead of COMPOSITION and REFERENCE if we find the distinction is not important to clients.
- **Self-contained nature of syntax makes it easy to ignore.**
  - At least for some legacy parser implementations

What’s Next?

- **Ensure all syntax constructs have clear examples.**
- **Continue implementation experiments.**
  - End-to-end (provider through client) application of “real world” cases
  - Show that client can “understand” certain models like Source, CAOM, etc.
  - Multiple providers/clients desirable
  - Curate a list of validations and continue development of validator(s)
- **Clean up volute to archive past approaches.**
- **Solicit/gather feedback.**
  - Make updates to syntax, document, examples and implementations.
- **Formalize syntax with VOTable 1.4 specification**
  - Done in time to allow review prior to next Interop
  - Are there other VOTable updates that *need* to go in at the same time?

References/Documentation

- **Mapping document working draft**
  - HTML version has scrollable examples: http://doc-ivoa.rhcloud.com/document/9458a3d669c93068f4290e0f58a3e058b18ad7ef
  - PDF: https://volute.g-vo.org/svn/trunk/projects/dm/vo-dml/doc/VO-DML_mapping_WD.pdf
- **Parser**
  - Documentation: https://volute.g-vo.org/svn/trunk/projects/dm/vo-dml/mapping/MAST%20VODML%20Parser/VODML%20Parser.html
  - Client: https://masttest.stsci.edu/vodml/Mashup/Clients/Mast/Portal.html
- **Example VOTables**
  - See several examples embedded in mapping document.
  - Catch all for most constructs (GL): https://volute.g-vo.org/svn/trunk/projects/dm/vo-dml/mapping/test5.votable-1.4.xml
  - Cube (MCD): See *_annotated.vot files in https://volute.g-vo.org/svn/trunk/projects/dm/CubeDM-1.0/examples/
  - Tesselation (LM): https://volute.g-vo.org/svn/trunk/projects/dm/vo-dml/models/tesselation/%20votable.annotatedRef.xml
- **Jovial**
  - https://github.com/olaurino/jovial

References/Documentation

Mapping project page:

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