Mapping VOTable Data on Data Models: Implementation Status and Prospect

Laurent Michel, Julien Abid, François Bonnarel, Mark Cresitello-Dittmar, Somia Floret, Gille Landais, Mireille Louys, Gregory Mantelet, François-Xavier Pineau, Jesus Salgado

Mapping VOTable Data on Data Models: Implementation Status and Prospect

Laurent Michel[1], Julien Abid[1], François Bonnarel[1], Mark Cresitello-Dittmar[2], Somia Floret[3], Gille Landais[1], Mireille Louys[1,4], Gregory Mantelet,[1] François-Xavier Pineau,[1] and Jesus Salgado,[5]

[1]*Universit é de Strasbourg, CNRS, Observatoire Astronomique de Strasbourg, UMR 7550, F-67000 Strasbourg, France;* laurent.michel@astro.unistra.fr

[2]*Center for Astrophysics | Harvard & Smithsonian, Boston, USA*

[3]*Université Technologique de Belfort-Montbeliard, France*

[4]*Université de Strasbourg, CNRS, ICube Laboratory, UMR 7357, F-67412 Illkirch Cedex*

[5]*SKA Observatory, Jodrell Bank, Lower Withington, Macclesfield, Cheshire SK11 9FT, UK*

Abstract. Model Instances in VOTables (MIVOT) is a VO standard that defines a syntax to map VOTable data to any data model serialised in VODML (Virtual Observatory Data Modeling Language). This annotation schema operates as a bridge between data and models. It associates both VOTable metadata and data to model elements (class, attributes, types, etc.). It also brings up VOTable data or metadata that were possibly missing in the table, e.g., accurate description of a coordinate system or curation tracing. MIVOT became an IVOA recommendation in June 2023. Having this standard was necessary to exercise data models against real data and to make the data interpretation easier by using code based with shared models. This paper presents our on-going developments : reading and writing MIVOT annotations with a CDS RUST library, reading and interpreting annotations with AstroPy/PyVO and creating an add-on to the VOLLT TAP library able to annotate query responses on the fly.

1. Introduction

MIVOT (**?**) is an XML schema that defines a standard way to connect VOTable (**?**) data with VODML (**?**) compliant data models. The MIVOT syntax reproduces the structure of the mapped classes such as sky positions from *Meas/Coords* models (**?**, **?**) or fluxes from the *Mango* draft model [1]. It uses *@ref* XML attributes to establish connections between model leaves and columns of the data tables.
Producing instances of the classes outlined in the MIVOT block is fairly straightforward: simply replicate the class description (a compound XML element) and resolve

[1]https://github.com/ivoa-std/MANGO

the pending references with the data cells of the current row. Annotating query responses with MIVOT offers the following advantages:

1. The model views generated that way do not depend on the data server since the models are shared by all stakeholders.

2. The MIVOT syntax allows to connect data to each other e.g., associate measures with flags or errors.

3. MIVOT can provide VOTables with possibly missing metadata (coordinate system details, curation tracking, filter descriptions, flags, link semantics) defined in mapped models.

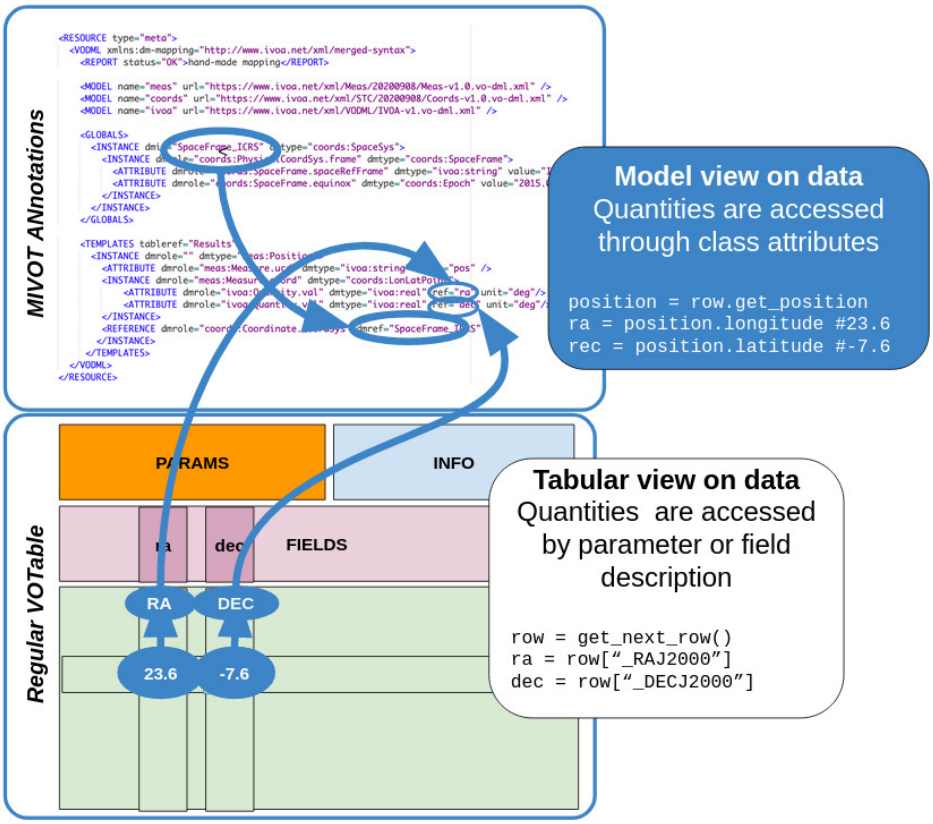

Figure 1. This figure shows how MIVOT binds model leaves with table columns. This mechanism allows client code to easily obtain a model view on the active data row.

2. Using MIVOT in the context of a TAP service

The goal here is to enable a TAP (**?**) server to annotate query responses on the fly. As it was outlined after the ADASS BoF 2021 (**?**), the requirements hold the following:

- The client application must be able to discover services able to annotate responses. This can be achieved by either a registry capability or a new TAP parameter specific for that purpose.
- The client application must be able to ask the server to generate MIVOT annotations for a given model.
- The query engine must be able to analyse the ADQL query to determine whether the selected columns can be mapped on the requested data model.
- The annotation engine must be able to retrieve MIVOT serializations of the model components that can be mapped.
- The annotation engine must understand the binding between model leaves and table columns to complete the annotation process.

3. Client Side

We have defined different API levels that can apply to any language (Python or RUST in our case).

- Level 0: extract the MIVOT block from the VOTable: No reference is resolved. The MIVOT block is made available as a parser subset. Level 0 has been implemented in Astropy 6.0.
- Level 1: provide access to a parser tree whose structure matches the model view of the current row. The internal references (e.g. association with coordinate systems) have been resolved. The attribute values have been set with the actual table data. This parser subset can be used for building whatever objects.
- Level 2: entail a few methods to ease the browsing of the level 1 output. Level 2 API allows to retrieve MIVOT elements by either class roles or data types. At this level, the MIVOT block still needs to be handled as parsed XML elements.
- Level 3: transform the layer 1 output into a dynamic object (in Python). Different components can be accessed through object paths. The field names of this object are derived from the roles played by the attributes in the model context.
- Level 4 (Python): build Astropy objects (e.g., SkyCoord) from the level 1 output.

Both of our implementations have a similar architecture. Level 0 is embedded in the VOTable parser and all further levels are implemented in a VO aware package.

4. Status and Prospect

The presented work is still in progress at the time of writing. The PyVO implementation is working up to layer 3. Current developments are guided by IVOA's request for a high-performance demonstrator processing celestial positions with epoch propagation (position, proper motion, parallax and radial velocity including correlated errors). On server side, our implementation is based on a former prototype (**?**) that led to an online

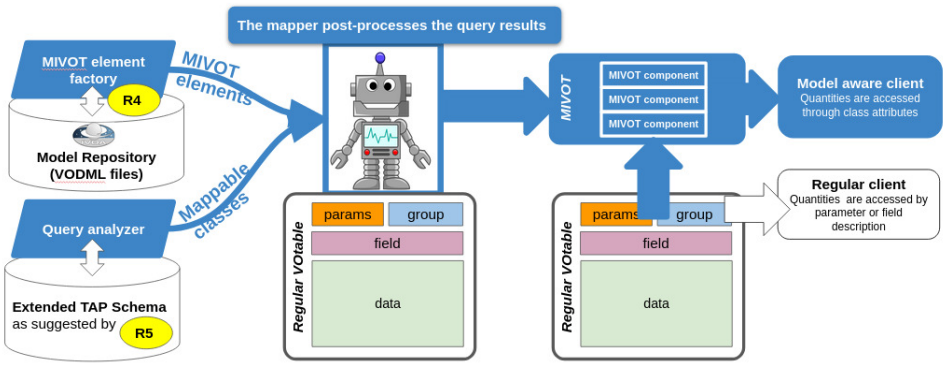

Figure 2. On the fly annotation by a TAP server: The annotator module knows the binding of the searched table FIELDS with the model leaves. It searches the MIVOT elements matching the classes to be mapped, sets the appropriate references in the MIVOT ATTRIBUTE, and packs all of these snippets on top of the VOTable data table.

demonstrator *xtapdb* [2] operated at Strasbourg by the Survey Science Consortium of XMM-Newton (SSC).

Acknowledgments. We would like to thank M. Marchand (CDS), B. Sipöcz, P.L. Lim and T. Donaldson (STSCI) for introducing us as AstroPy/PyVO contributors and for their careful review of our PRs. We would also like to thank the CDS, the CNES and the SSC for funding and supporting the internships that made this possible.

[2] `https://xcatdb.unistra.fr/xtapdb`