Data Model Workshop

Pre-interop Session

Mark Cresitello-Dittmar

May. 17, 2021

Introduction

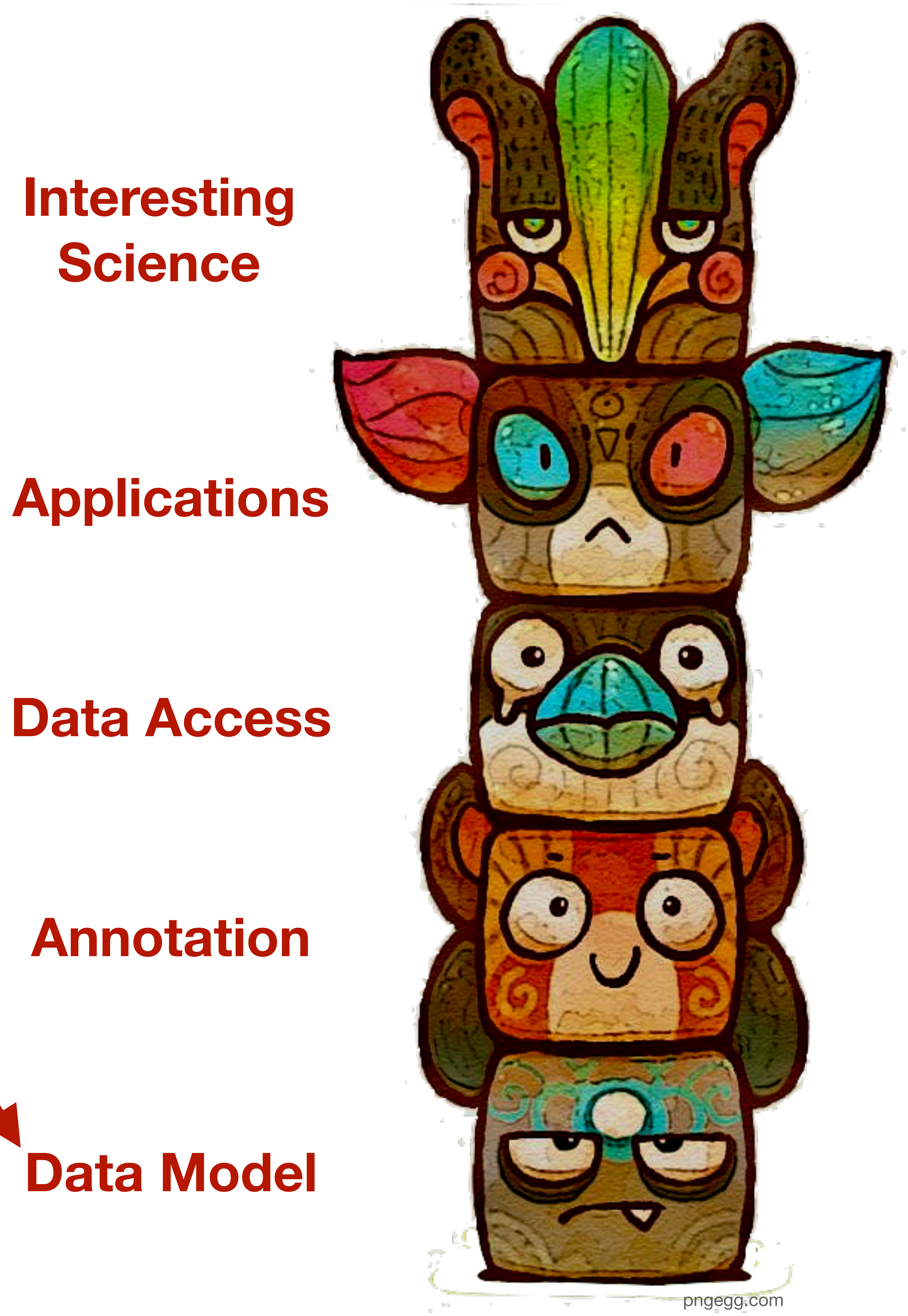
- Workshop Goals
    - Exercise models-in-progress in real world usage, on real world data
    - Exercise ability of annotation syntax proposals to map existing datasets to model instances
    - Demonstrate compatibility with common existing software (e.g. Astropy)
    - Demonstrate the potential for supporting “Interesting Science”

IVOA Hierarchy

Model Implementation Challenge

- The data models support several aspects of IVOA interest.

- They inform all users of the entities involved, their relations and associations to other entities.



Interesting Science

Applications

Data Access

Annotation

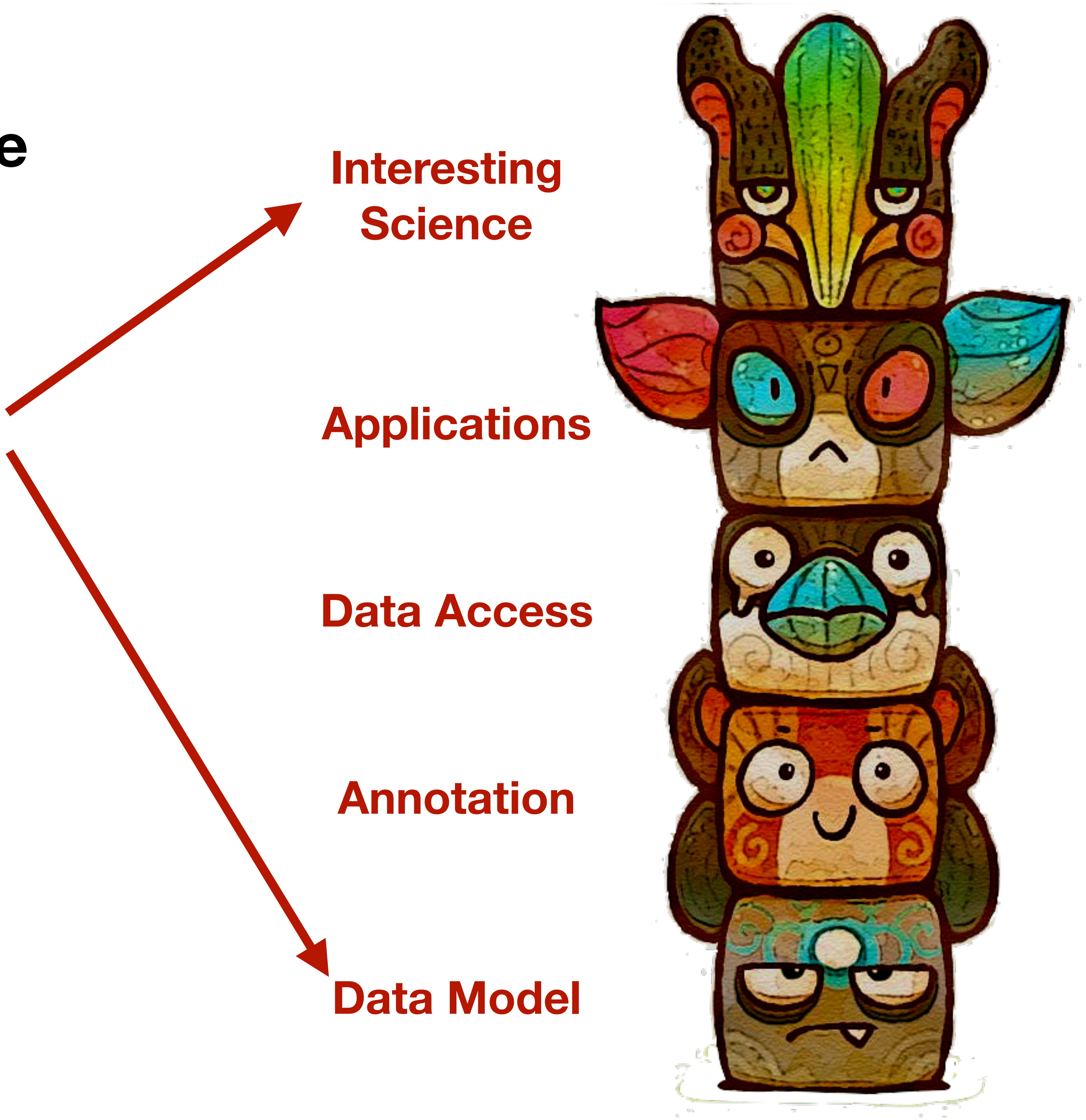
Data Model

pngegg.com

IVOA Hierarchy

Model Implementation Challenge

- Want to demonstrate that the models can actually support the Interesting Science cases..
- This is a significant commitment in resources to 'test' a model.



Interesting Science

Applications

Data Access

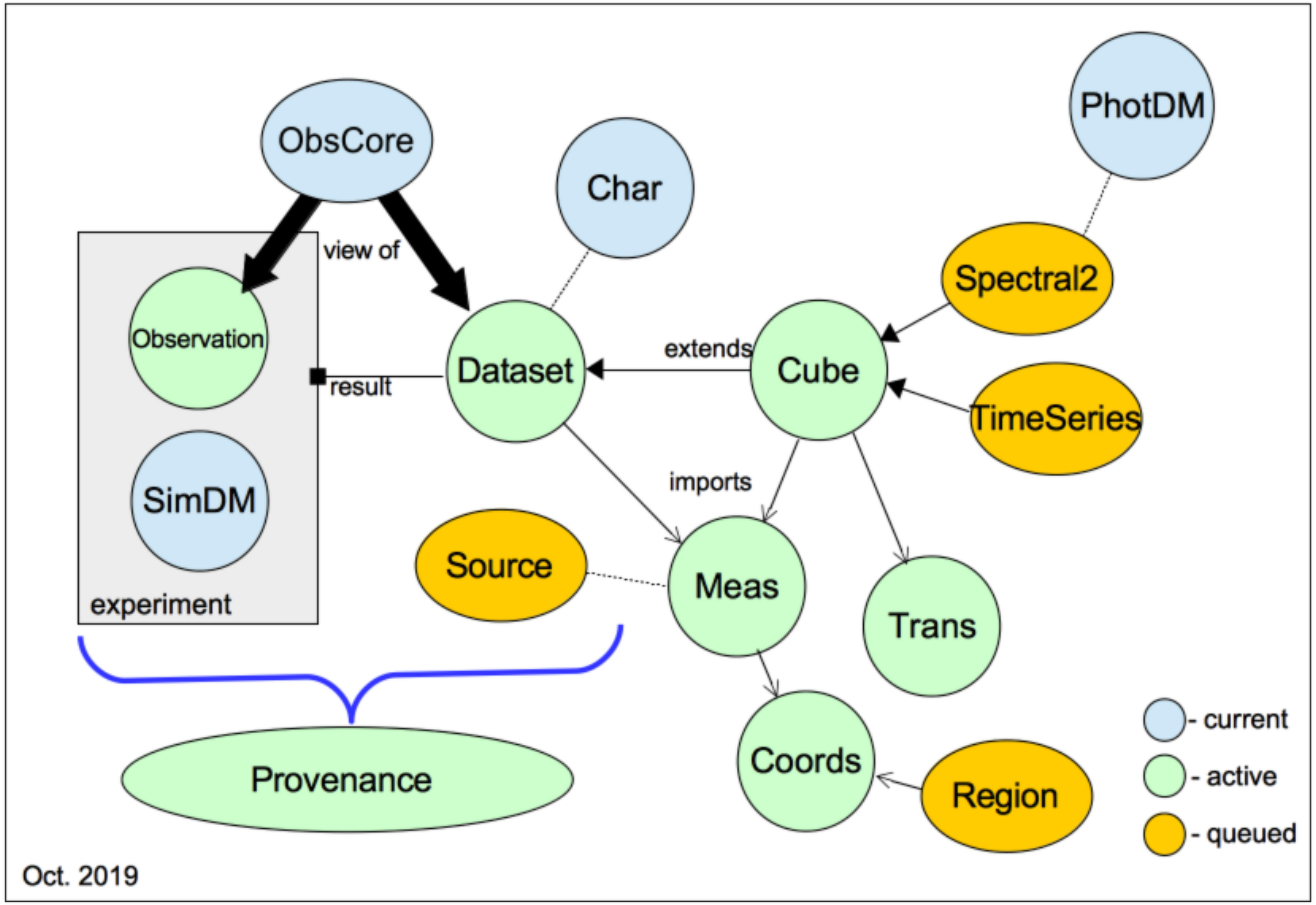
Annotation

Data Model

Models Used

Model Landscape

- Set of small, building block models used to construct complex data structures.



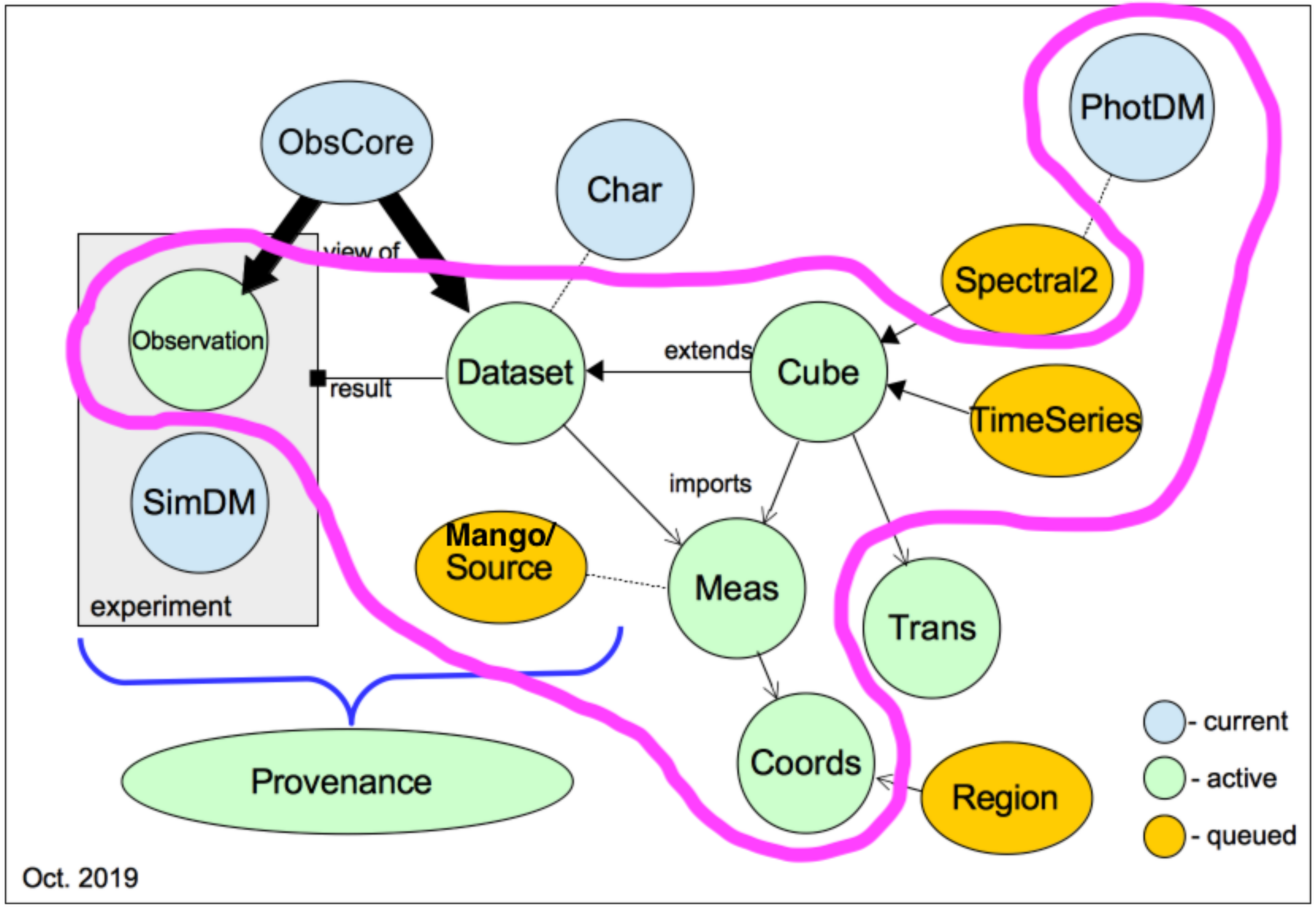
Oct. 2019

DM Working Group Twiki

Models Used

Model Landscape

- Set of small, building block models used to construct complex data structures.
- Models used in workshop cases



Oct. 2019

DM Working Group Twiki

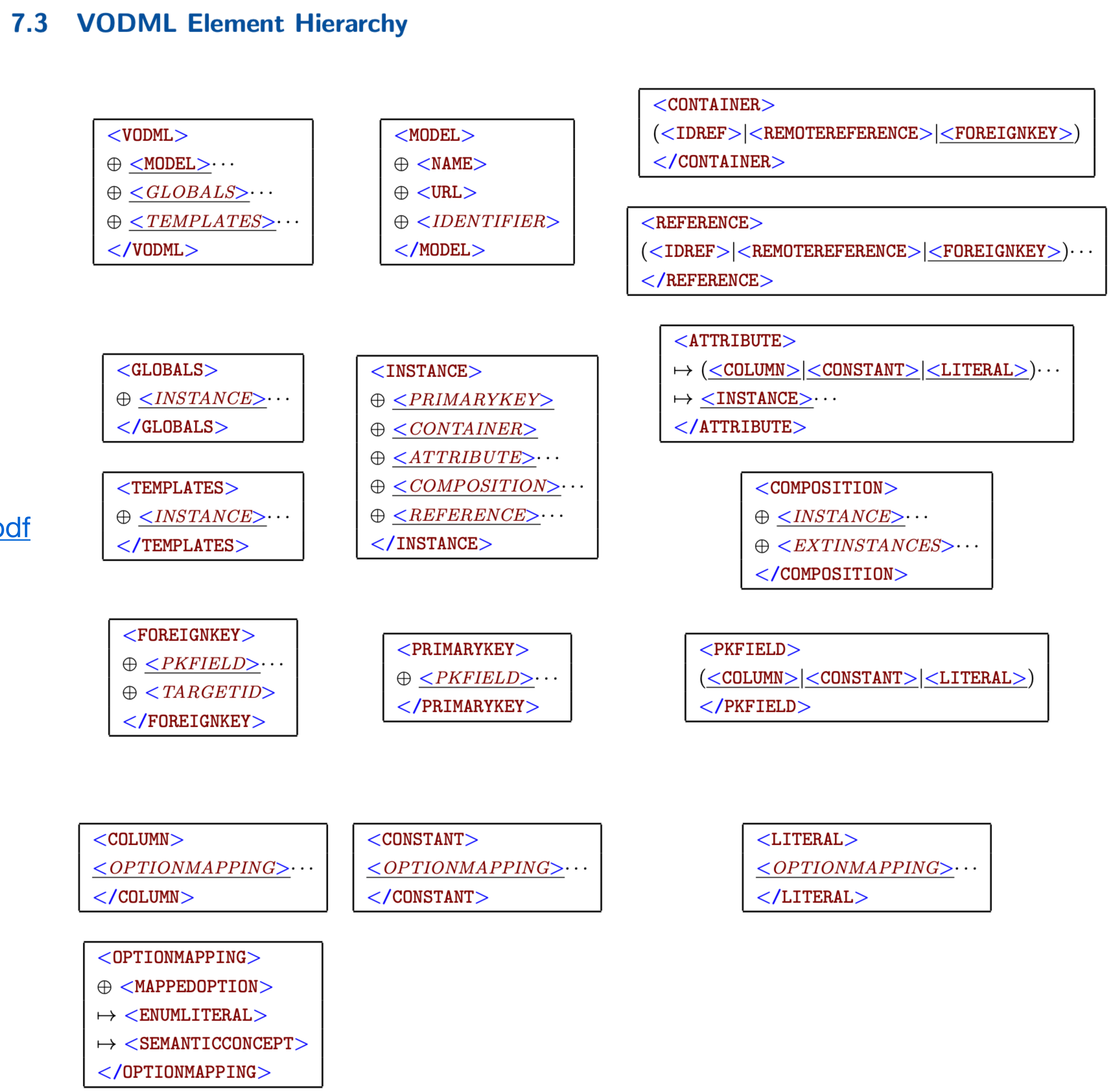
Annotation

VO-DML Mapping Syntax

“Mapping Data Models to VOTable”
(G. Lemson, O. Laurino et. al.)
Working Draft: 2017-03-23

https://volute.g-vo.org/svn/trunk/projects/dm/vo-dml-mapping/doc/VO-DML_mapping_WD.pdf

https://github.com/ivoa/mapping-vodml



7.3 VODML Element Hierarchy

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

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

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

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

<GLOBALS>
⊕ <INSTANCE>···
</GLOBALS>

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

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

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

<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>
```

Annotation

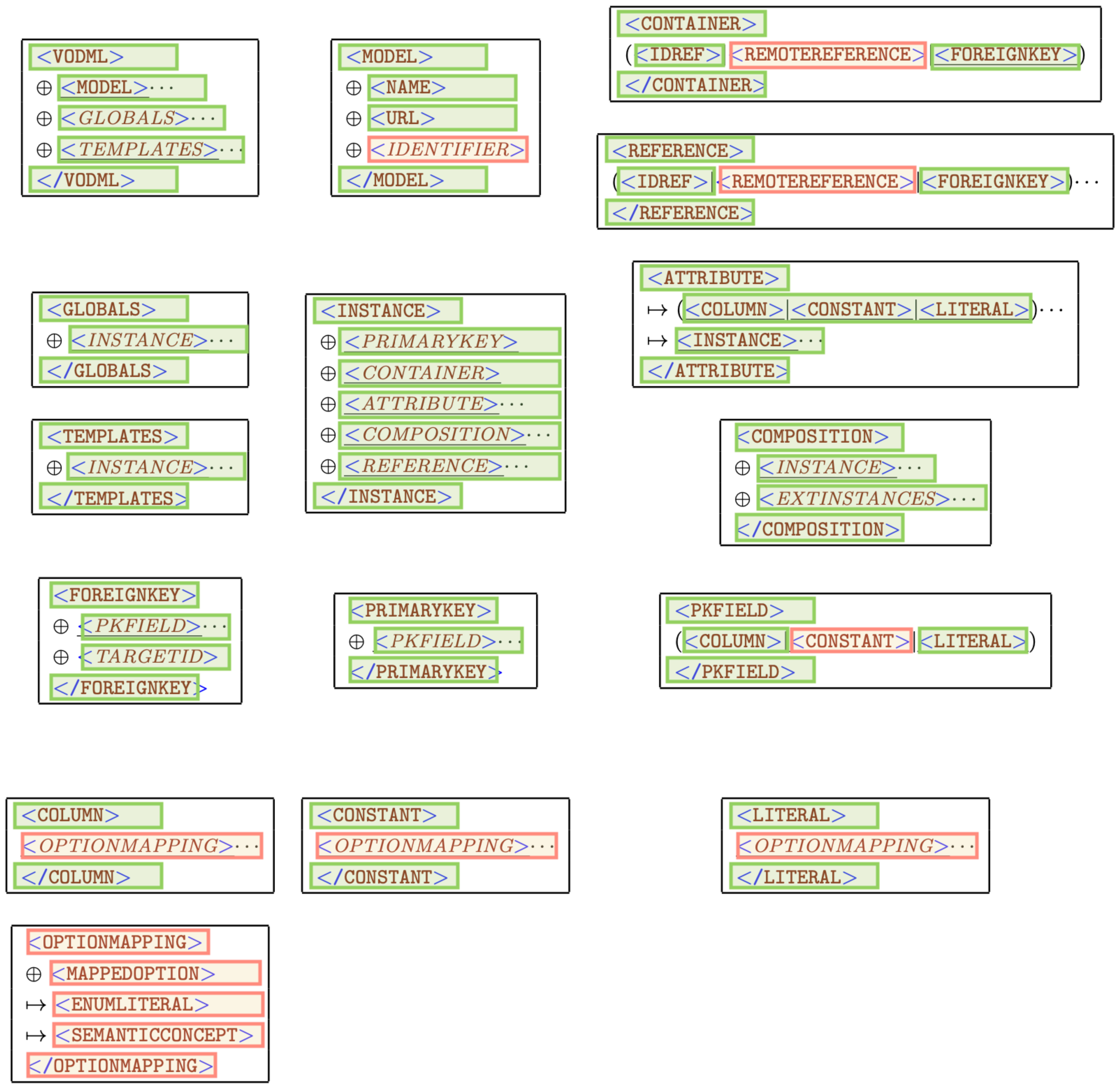
VO-DML Mapping Syntax

“Mapping Data Models to VOTable”
(G. Lemson, O. Laurino et. al.)
Working Draft: 2017-03-23

https://volute.g-vo.org/svn/trunk/projects/dm/vo-dml-mapping/doc/VO-DML_mapping_WD.pdf

https://github.com/ivoa/mapping-vodml

7.3 VODML Element Hierarchy



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

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

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

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

<GLOBALS>
⊕ <INSTANCE>···
</GLOBALS>

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

<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>
```

Software

Packages used in workshop implementations

- **Jovial - (Java)**
    - Developed by Omar Laurino; updated by me to the current data model content.
    - Generates annotation from DSL representation of instances.
- **Rama - (Python)**
    - Developed by Omar Laurino; updated by me to the current data model content and bug fixes/enhancements.
    - Parses annotation to generate instances of VO Data Model Classes.
    - Attaches adapters which translate certain VO Data Model Classes to corresponding Astropy types (with complete coordinate system specs).
        - eg: meas:Point.coord -> astropy:SkyCoord
        - eg: meas:Time.coord -> astropy:Time
- **Astropy - (Python)**
    - Unit conversions, Coordinate system conversions, epoch migration
    - Units/Quantity, SkyCoord, Time packages
- **MatPlotLib - (Python)**
    - Generate plots

Case 1

Column Grouping

- **Description:** Exercise ‘Associated Parameters’ feature of Mango model. Property A is ‘in some way’ related to other Properties.
- **Data:** Vizier dataset
- **Challenges:**
  - Annotate Source with radial velocity property
  - Associate radial velocity property with columns/‘properties’
    - Quality ‘grade’
    - #plates used to determine RV value
    - Observatory code
- **Models:**
  - Mango, Measurements, Coordinates
- **Results:**
  - GitHub Implementation Page shows the annotated and associated properties, but the properties are empty as I don’t believe these items fit under the Measurement umbrella.
    - This case will inform Mango and Measurement model development going forward.

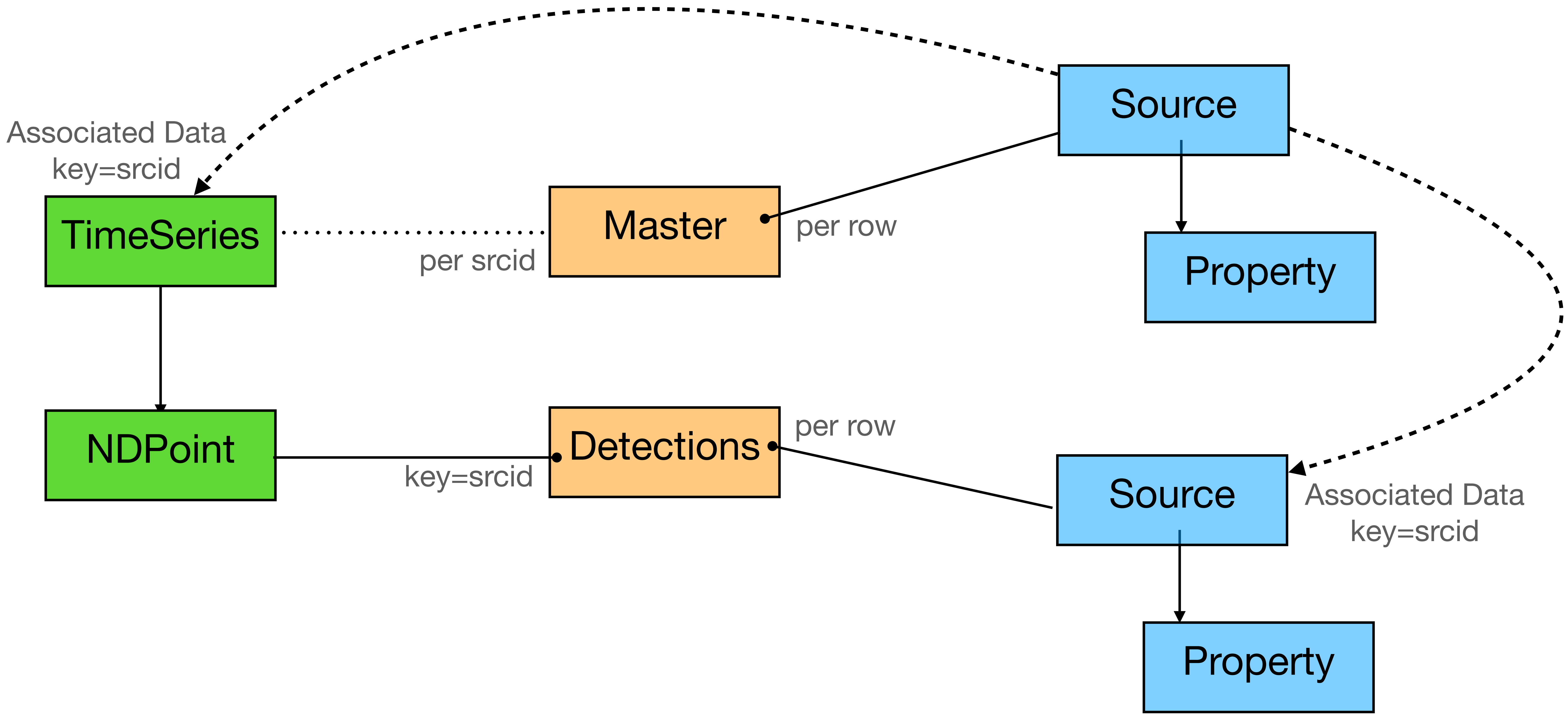
Case 2
Combined Data

- **Description:** Multiple Table annotation challenge; annotating to multiple models, exercise associated data feature of Mango model
- **Data:**
    - 4XMM - Table with ‘Source’ properties + Table with 1 or more links to Spectra for particular sources
    - CSC - Master Source Table (one record per source) + Detections Table (one record per observation)
- **Challenges (CSC example):**
    - Annotate to Mango and Cube (TimeSeries as Cube) models
        - Annotate Master table to Mango model (Source)
            - Associate properties from same table
            - Associate Detection Source instances from Detections table for each source
            - Associate TimeSeries derived from Detection table to each source
        - Annotate Detections table to Mango model
        - Annotate SparseCube (TS) for each Master Source, whose data is populated from the Detections table.
- **Models:**
    - Mango, Cube, Dataset, Measurements, Coordinates, PhotDM

Case 2

Combined Data

Code: GitHub Implementation Page



Case 3a

Standard Properties

- **Description:** Easily ‘find’ scientifically relevant properties.
- **Data:** 4XMM, CSC2, GAIA
- **Challenges:**
  - Annotate properties in each file
    - Obs. Time, Obs. Duration, Position, Photometry, Hardness Ratios, Flags (detection, variability, quality)
    - Informs the measurement model extension process/requirements
  - Use the same script to locate/extract property data from each file
- **Models:**
  - Mango, Measurements, Coordinates, PhotDm

Case 3a

Standard Properties

Code GitHub Implementation Page

```
# Load annotated file
doc = Reader( Votable(infile) )

# Extract list of Source records
#  - Source model provides structure, organizing the Properties
catalog = doc.find_instances(Source)[0]

sys.stdout.write("\n")
sys.stdout.write("o Goal: High Level content summary\n")
sys.stdout.write("    o Number of records: %d\n"%( len(catalog.identifier) ) )
sys.stdout.write("    o Number of unique Sources: %d\n"%( len(set(catalog.identifier)) ) )

# Summarize content of example Source record.
srcno = 2
source = catalog.unroll()[srcno]

sys.stdout.write("\n")
sys.stdout.write("o Goal: Detail Level content summary\n")
sys.stdout.write("    o Source number: {}\n".format( srcno+1 ) )
sys.stdout.write("    o Identifier: {}\n".format( source.identifier ))

for prop in ( source.parameter_dock ):
    sys.stdout.write( "    o Property: semantic={}, ucd={}\n".format( prop.semantic.label, prop.ucd ))
    sys.stdout.write( "        o {}\n".format( measure_toString( prop.measure )))
```

* Script will work on ANY file annotated to the model(s)
* Generic scan of properties, or can target specific properties directly.
* Easily identify common properties
* Easy access to associated metadata (frames, bands, etc)
* Use Astropy packages (or other) to manipulate data to common basis or work an interesting science thread.

Chandra Catalog Results

```
• Goal: High Level content summary
    ○ Number of records: 1000
    ○ Number of unique Sources: 326
• Goal: Detail Level content summary
    ○ Source number: 3
    ○ Identifier: 2CXO J104732.7+123024
    ○ Property: semantic=position, ucd=pos
        ▪ Position: ( 233.542479 [deg], 57.535140 [deg] ) [GALACTIC]
    ○ Property: semantic=flux, ucd=phot.flux
        ▪ Photometry: (9.743e-15 [erg/s/cm^2]) [band=CHANDRA/ACIS.broad]
    ○ Property: semantic=hardness_ratio, ucd=phot.color
        ▪ HardnessRatio: 0.239 range(low: 0.028, high: 0.439) [band_low: CHANDRA/ACIS.hard, band_high: CHANDRA/ACIS.soft]
    ○ Property: semantic=hardness_ratio, ucd=phot.color
        ▪ HardnessRatio: 0.311 range(low: 0.132, high: 0.489) [band_low: CHANDRA/ACIS.medium, band_high: CHANDRA/ACIS.soft]
    ○ Property: semantic=hardness_ratio, ucd=phot.color
        ▪ HardnessRatio: -0.080 range(low:-0.242, high: 0.077) [band_low: CHANDRA/ACIS.medium, band_high: CHANDRA/ACIS.soft]
    ○ Property: semantic=obs.start, ucd=time
        ▪ Time: 2006-04-09T10:51:35.000 [TT]
    ○ Property: semantic=quality, ucd=src.extent
        ▪ Flag: 0 [Not Extended]
    ○ Property: semantic=quality, ucd=src.var
        ▪ Flag: 1 [Source hardness ratios are statistically inconsistent between two or more observations]
```

Case 3b

Proper Motion

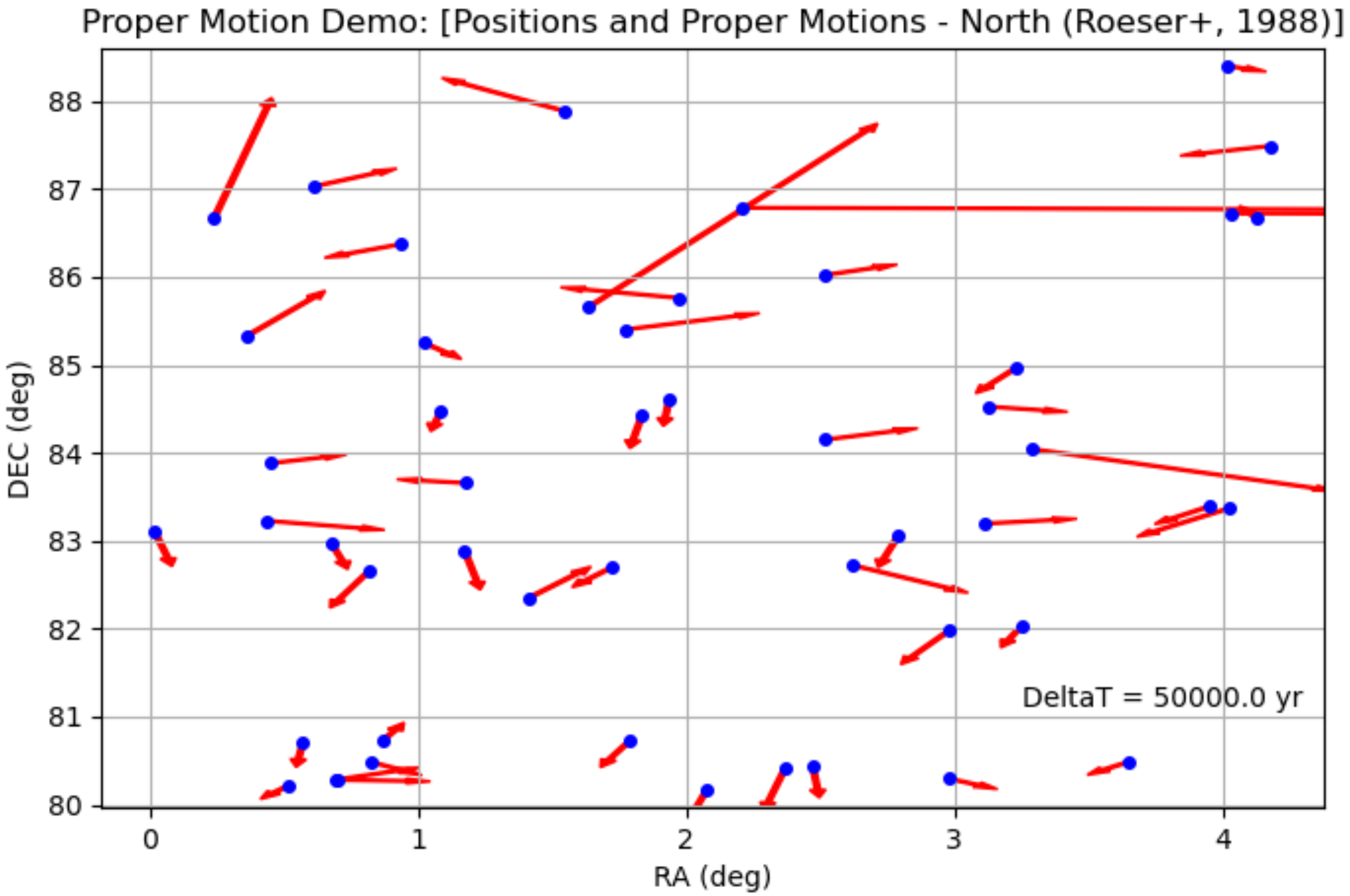
- **Description:** Proper Motion ‘Slider’
- **Data:** Vizier dataset with position and proper motion data
- **Challenges:**
  - Identify and extract position and proper motion data
  - Associate them and illustrate relative motions of sources
- **Models:**
  - Measurements, Coordinates
    - Note: Intentionally annotated ONLY to these models. The implementation script would work, unchanged, if these were within the context of a Data Product (e.g. Source or Cube)

Case 3b

Proper Motions

Code: GitHub Implementation Page

```
doc = Reader( Votable(infile) )
pos = doc.find_instances(Position)[0]
pm  = doc.find_instances(ProperMotion)[0]

# Setup plot
fig = plt.figure(figsize=[8.0,5.0])
ax = fig.add_subplot(111)
ax.grid(True)
ax.set_title("Proper Motion Demo: [Positions and Proper Motions - North (Roeser+, 1988)]")
ax.set_xlabel("RA ({})".format(pos.coord.ra.unit))
ax.set_ylabel("DEC ({})".format(pos.coord.dec.unit))
ax.set_xlim( np.min(pos.coord.ra.value)-0.2, np.max(pos.coord.ra.value)+0.2 )
ax.set_ylim( np.min(pos.coord.dec.value)-0.2, np.max(pos.coord.dec.value)+0.2 )

# Gather data and plot
xvals = pos.coord.ra.value
yvals = pos.coord.dec.value

# Determine offsets due to proper motion
deltaT = (50000.0 * u.Unit('yr'))
dx     = (pm.lon.cval * deltaT).to(u.deg).value
dy     = (pm.lat.cval * deltaT).to(u.deg).value

# Plot Postions with Arrow indicating proper motion direction and speed
ax.plot( xvals, yvals, markersize=4, marker="o", linestyle='', color="blue" )
ax.text( 3.25, 81.1, "DeltaT = {}".format( deltaT ))
for n in range(len(xvals)):
    ax.arrow( xvals[n], yvals[n], dx[n], dy[n], width=0.02, color="red" )

plt.show()
```



- Easy/Automatic conversion to Astropy SkyCoord enables
  - Conversion of coordinate frame
  - Quantity math handles unit conversion and scaling by time
  - Use of its proper motion migration code.

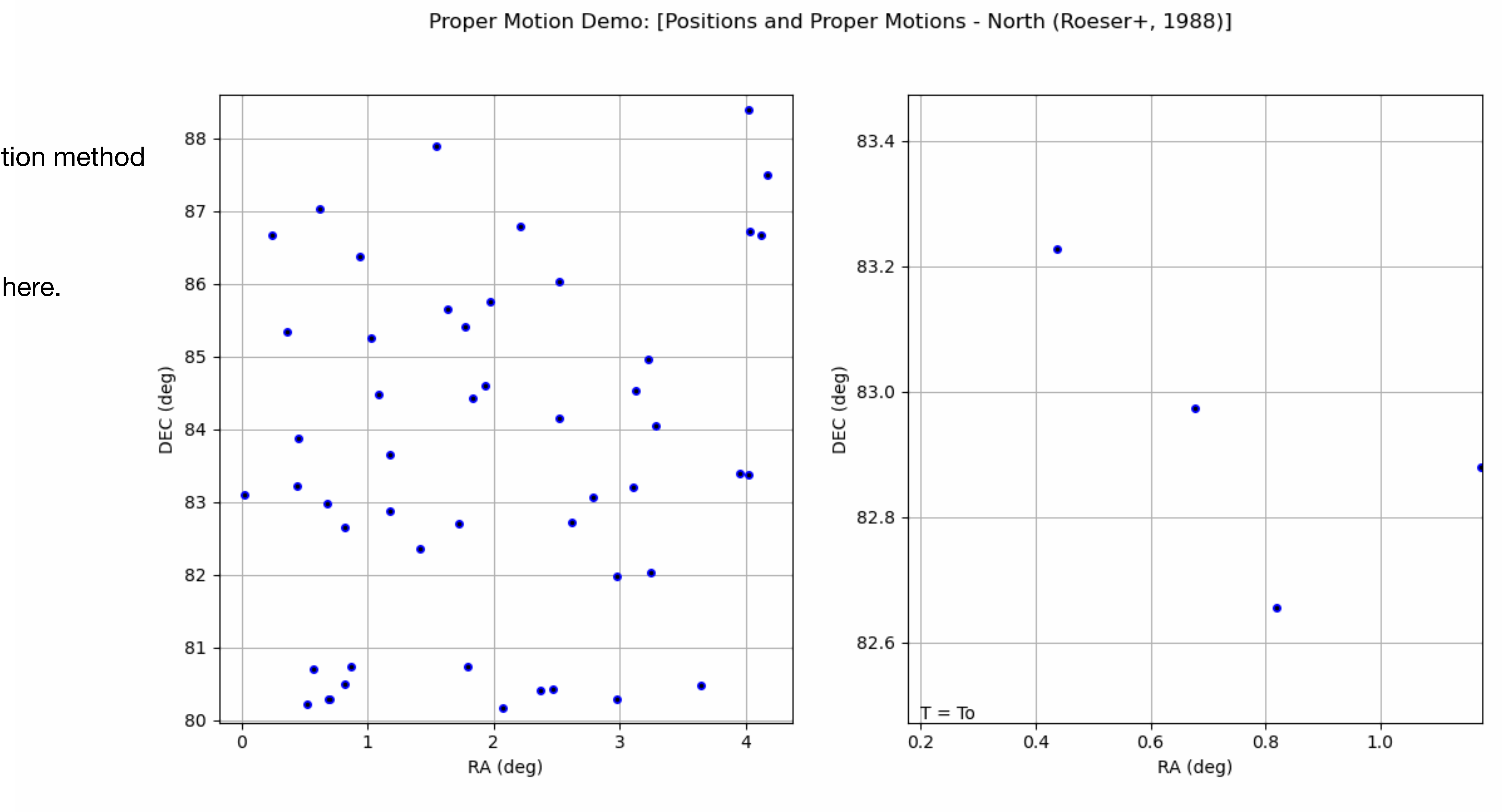
Case 3b

Proper Motion Slider

Animation: [Proper Motion Animation]

- Combines
  - Astropy SkyCoord apply_space_motion method
  - MatPlotLib FuncAnimation
- To propagate sources over time

Note: cosDec application info is important here.



Case 4

Time Series

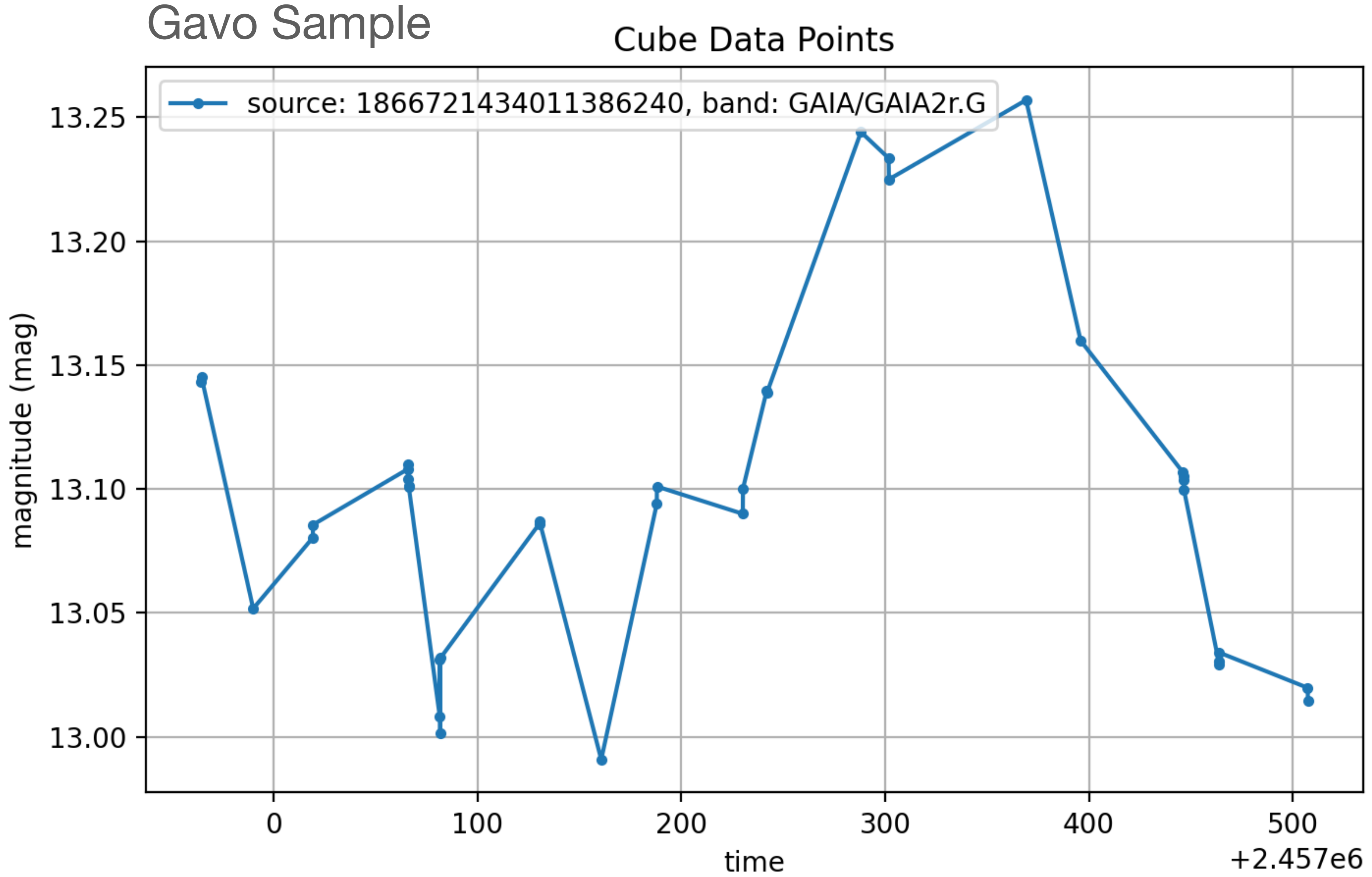
- **Description:** Identify Time Series instances
- **Data:**
  - Gavo - Simple time series table
  - ZTF - Time Series for each source in field of view
  - GAIA multi-band - Time Series using multiple filters, and multiple sources; compact native serialization
- **Challenges:**
  - Annotate datasets to Cube model (TimeSeries as Cube)
    - Identify ‘dependent’ and ‘independent’ axes
    - Associate data and errors
    - Plot TimeSeries data
  - Use the same script to process and plot each file. Even though the native representation is VERY different, the client sees the same view.
- **Models:**
  - Cube, Dataset, Measurements, Coordinates, Mango (meas extensions), PhotDM

Case 4

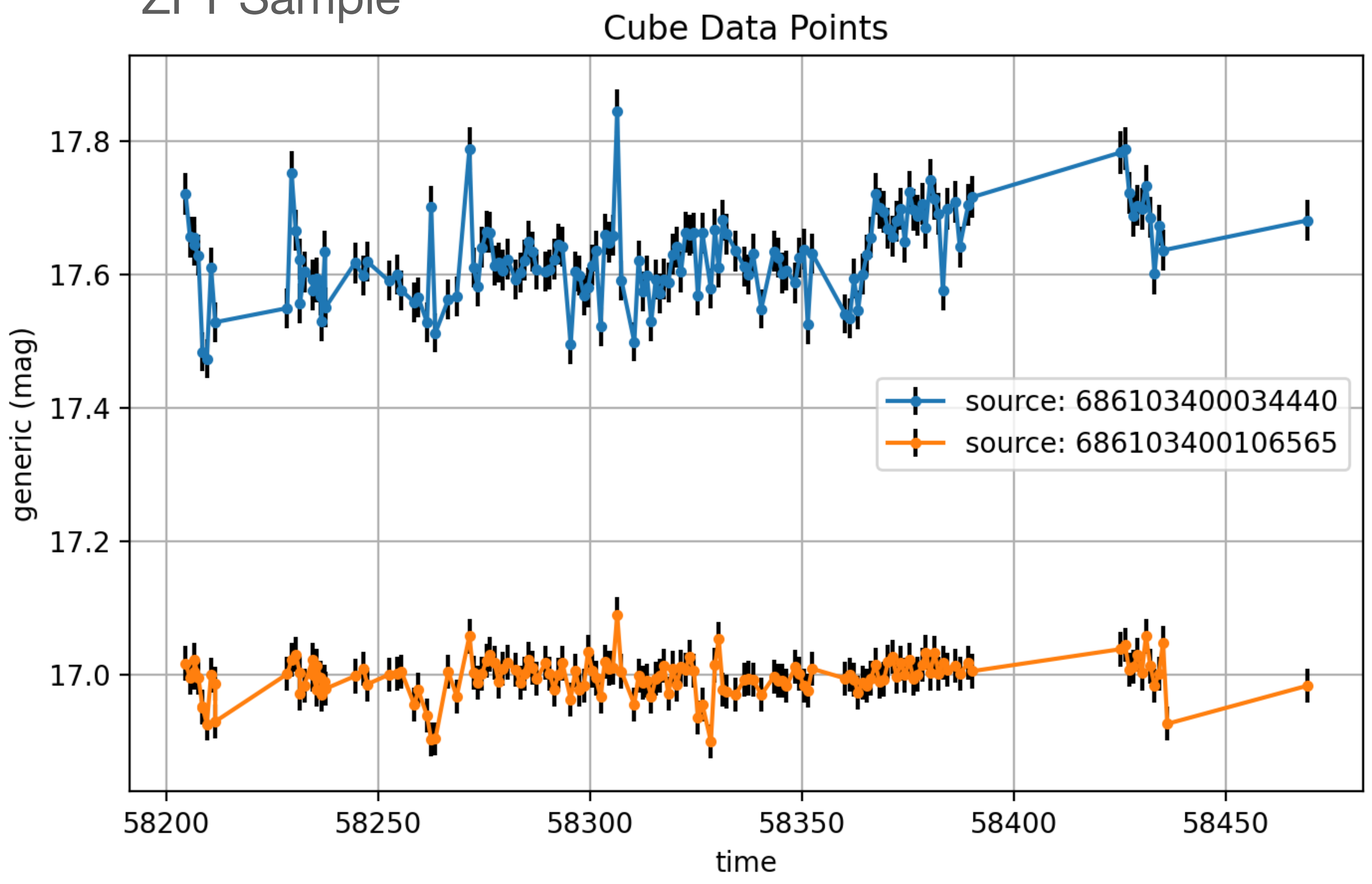
Time Series

Code: GitHub Implementation Page

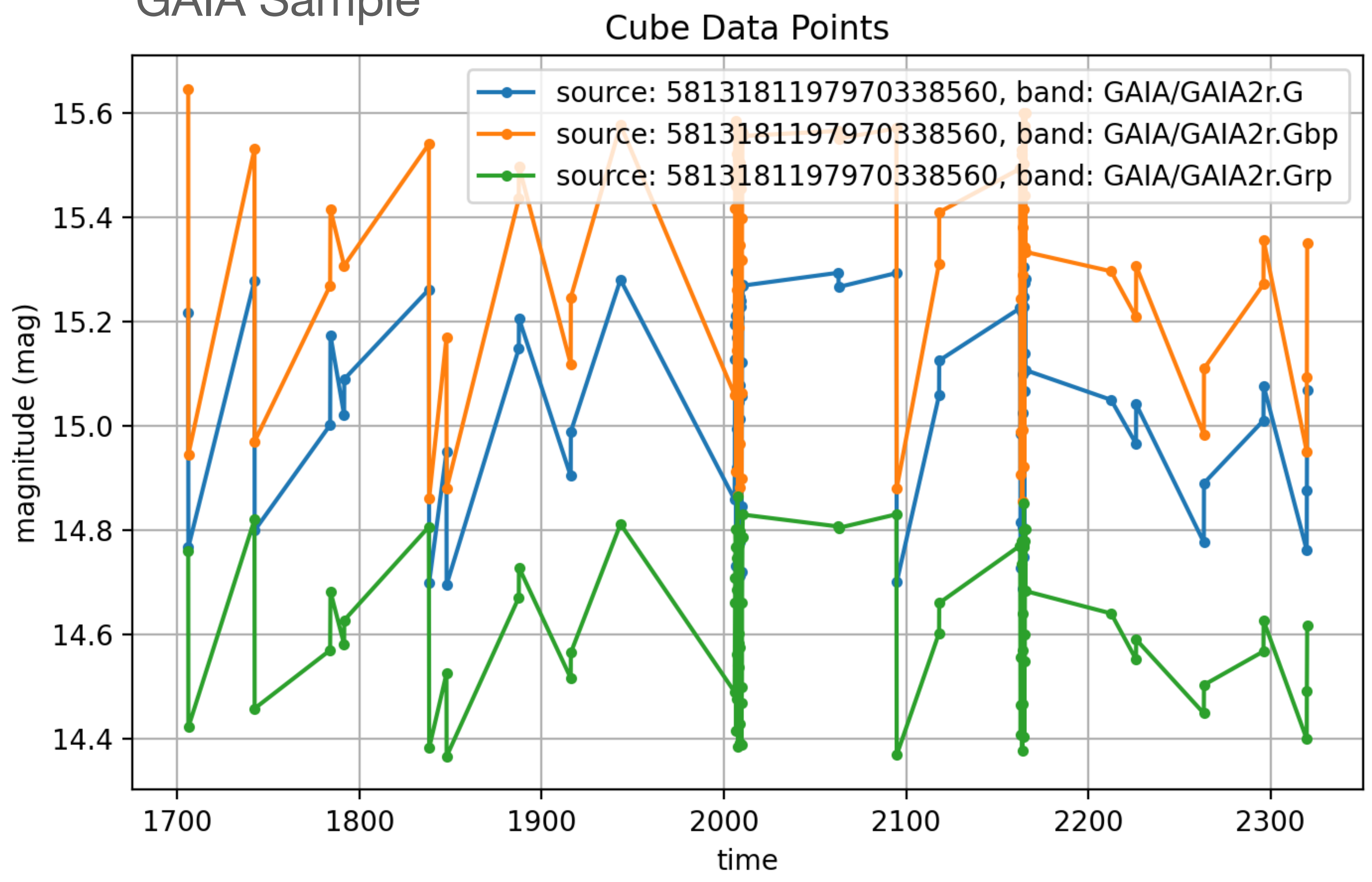
Gavo Sample



ZFT Sample



GAIA Sample



Conclusion

- Models
  - Models provided high level of support for the workshop cases from very simple to very complex
    - Core models: Identified a couple adjustments to make, but the framework is sound.
    - Good experience with extending core Measurements with different sorts of data.. (Photometry, Hardness Ratios)
      - and refining the line between a Measurement and other forms of data.. (Flags, Classifiers)
- Annotations
  - Mapping syntax supported ALL cases, from most simple case to compact GAIA multi-band TimeSeries.
  - Specifics of annotation not visible at user level
  - Some elements are more intuitive than others; gathered good experience to define final syntax
- Demonstrated potential
  - Core models are compatible with the Astropy internal model: easy conversion from Model instance to Astropy instance.
  - Proper Motion Slider: identify/combine Position and Proper Motion, unify coordinate system, migrate in Time.
  - Time Series: Using same code, extract and manipulate TimeSeries from multiple resource with VERY different underlying data structures.

Resource Summary

- Mapping Syntax
  - Work Draft Document
    - Volute: https://volute.g-vo.org/svn/trunk/projects/dm/vo-dml-mapping/doc/VO-DML_mapping_WD.pdf
    - Git: https://github.com/ivoa/mapping-vodml
- Jovial Library
  - Version used in this project: https://github.com/mcdittmar/jovial
  - Master repository: https://github.com/olaurino/jovial
- Rama module
  - Version used in this project: https://github.com/mcdittmar/rama
  - Master repository: https://github.com/olaurino/rama
- Workshop Implementations
  - Column Grouping: https://github.com/ivoa/dm-usecases/tree/main/usecases/column_grouping/mcd-implementation
  - Combined Data: https://github.com/ivoa/dm-usecases/tree/main/usecases/combined_data/mcd-implementation
  - Standard Properties: https://github.com/ivoa/dm-usecases/tree/main/usecases/standard_properties/mcd-implementation
  - Time Series: https://github.com/ivoa/dm-usecases/tree/main/usecases/time-series/mcd-implementation