
# HSC Workflow Using VO-DML

Tom Donaldson

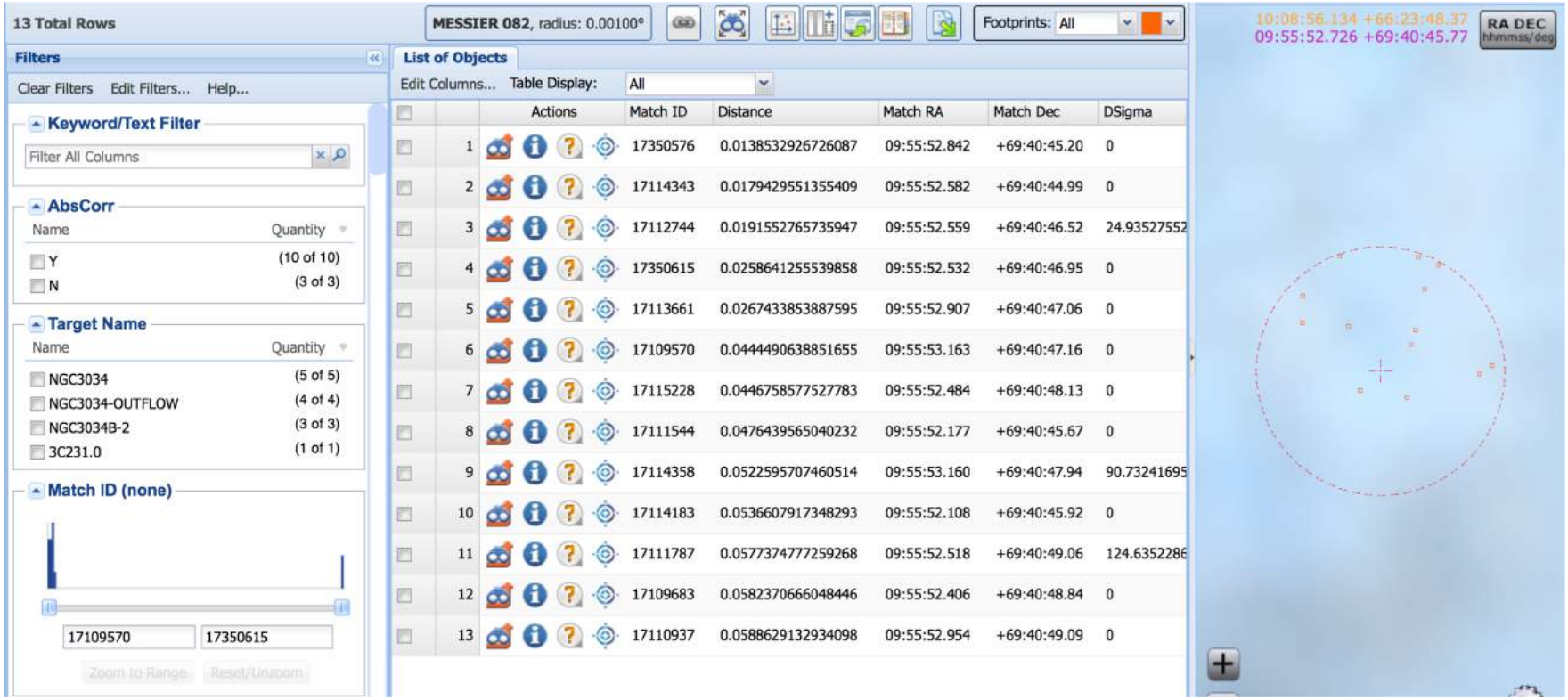
IVOA Interop

Northern Spring, 2018


Example Science Workflow for Hubble Source Catalog

# Perform a Hubble Source Catalog Cone Search

- Results are “matches”, i.e., a cross-correlation of one or more source detections.



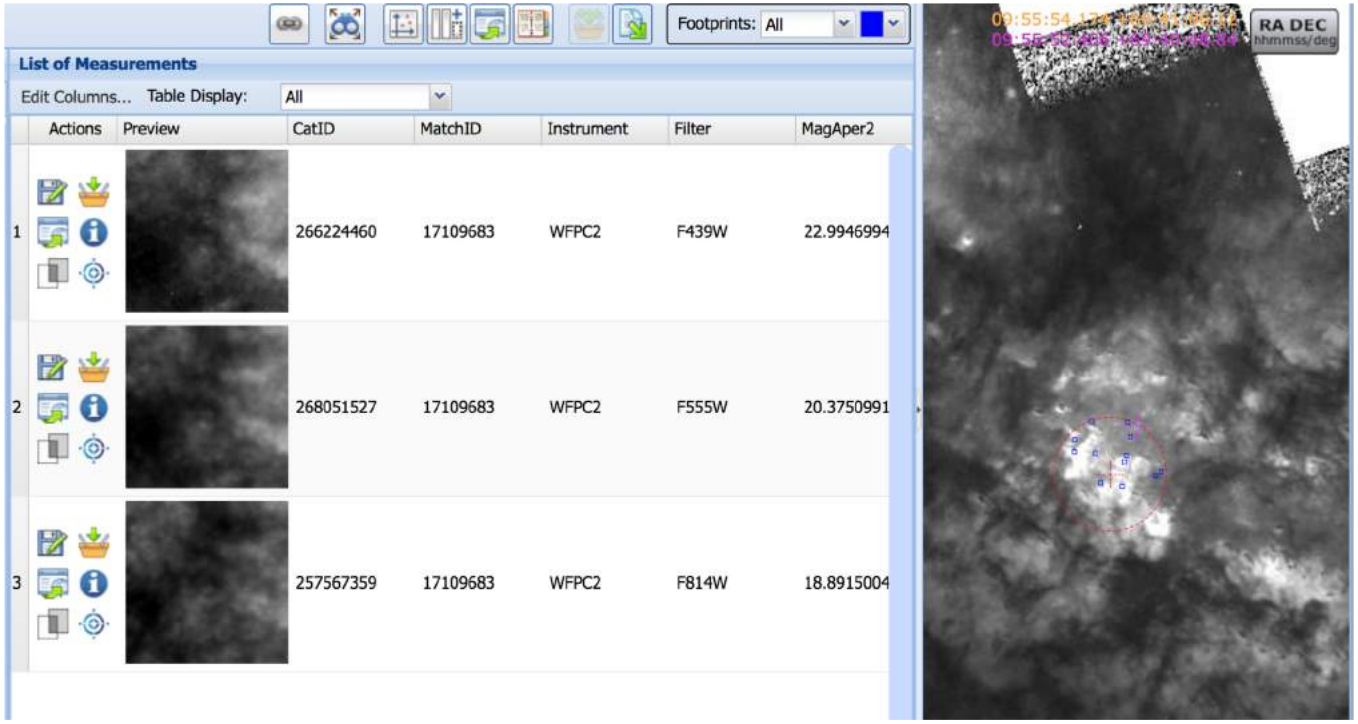
## Choose a match of interest

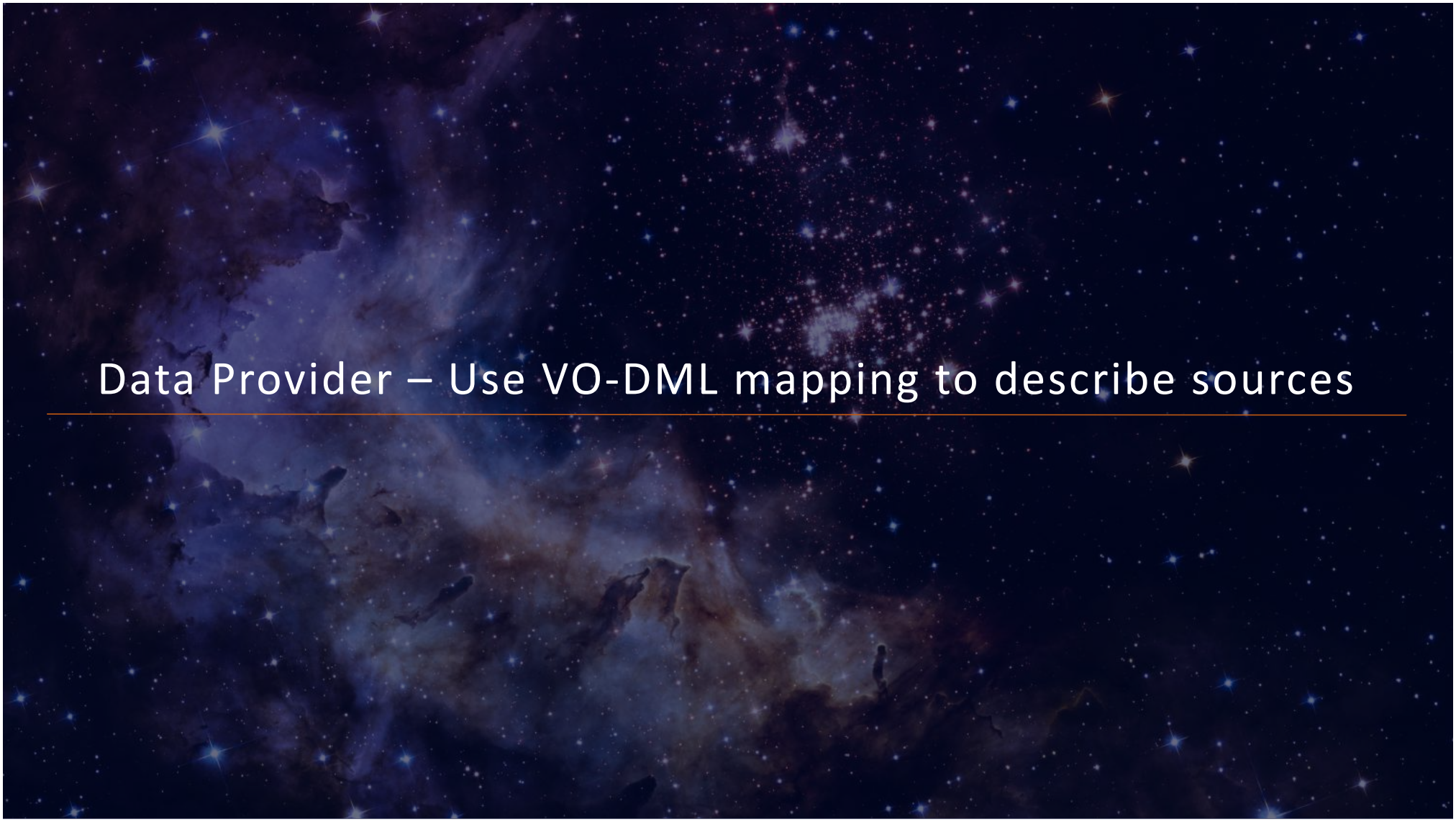
- Of matches observed in multiple filters, choose one with brightest B magnitude.

| | Actions | Match ID | Match RA | Match Dec | Num Filters | W2_F336W | W2_F439W | W2_F502N | W2_F547M | W2_ |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | 17112744 | 09:55:52.559 | +69:40:46.52 | 5 | NaN | NaN | 19.60960006... | 19.14710044... | NaN |
| 2 | | 17114358 | 09:55:53.160 | +69:40:47.94 | 5 | NaN | NaN | 19.21360015... | 18.94580078... | NaN |
| 3 | | 17111787 | 09:55:52.518 | +69:40:49.06 | 5 | NaN | NaN | NaN | 20.57239913... | 21.6 |
| 4 | | 17109683 | 09:55:52.406 | +69:40:48.84 | 3 | NaN | 22.99469947... | NaN | NaN | 20.3 |
| 5 | | 17115228 | 09:55:52.484 | +69:40:48.13 | 2 | NaN | 23.68339920... | NaN | NaN | 21.4 |
| 6 | | 17114183 | 09:55:52.108 | +69:40:45.92 | 2 | NaN | NaN | NaN | NaN | NaN |
| 7 | | 17350576 | 09:55:52.842 | +69:40:45.20 | 1 | 20.76779937... | NaN | NaN | NaN | NaN |
| 8 | | 17114343 | 09:55:52.582 | +69:40:44.99 | 1 | 20.39450073... | NaN | NaN | NaN | NaN |
| 9 | | 17350615 | 09:55:52.532 | +69:40:46.95 | 1 | 21.71129989... | NaN | NaN | NaN | NaN |
| 10 | | 17113661 | 09:55:52.907 | +69:40:47.06 | 1 | NaN | NaN | NaN | NaN | NaN |
| 11 | | 17109570 | 09:55:53.163 | +69:40:47.16 | 1 | 20.52269935... | NaN | NaN | NaN | NaN |
| 12 | | 17111544 | 09:55:52.177 | +69:40:45.67 | 1 | NaN | NaN | NaN | NaN | NaN |
| 13 | | 17110937 | 09:55:52.954 | +69:40:49.09 | 1 | 20.83860015... | NaN | NaN | NaN | NaN |

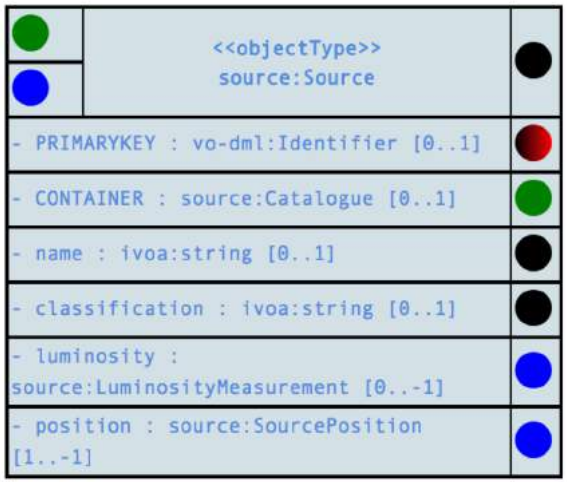
# Load all the detections for chosen match

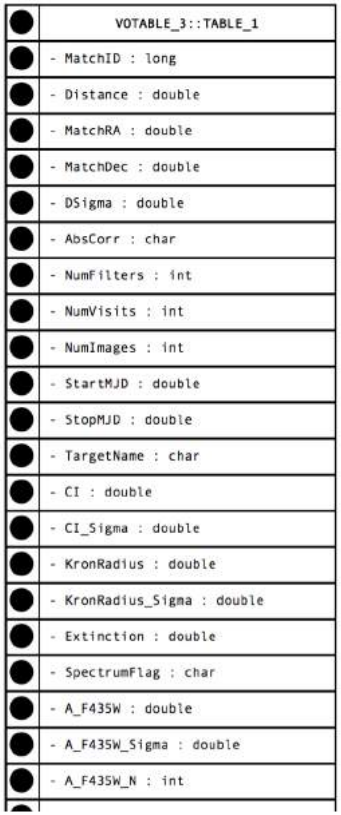
- Each detection was extracted from an image.
- Cutouts around match are shown.



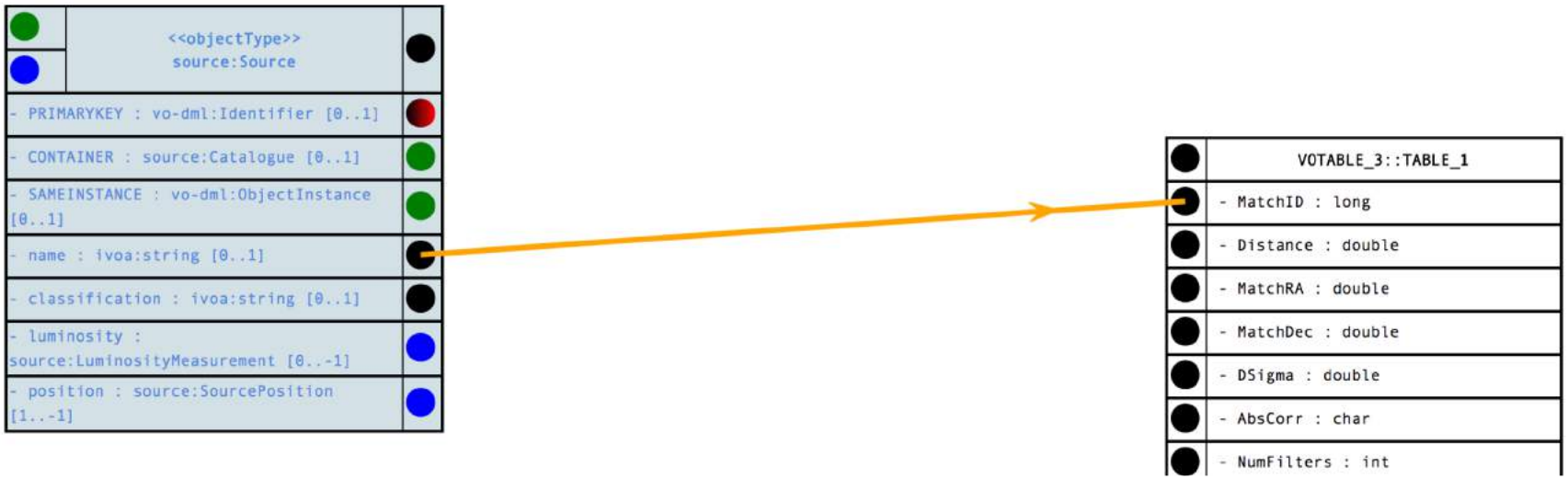

Data Provider – Use VO-DML mapping to describe sources

# Map Source model instances into the HSC VOTable

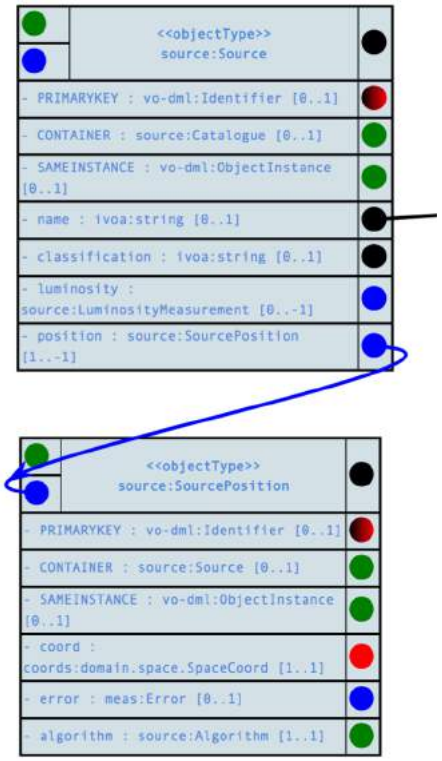


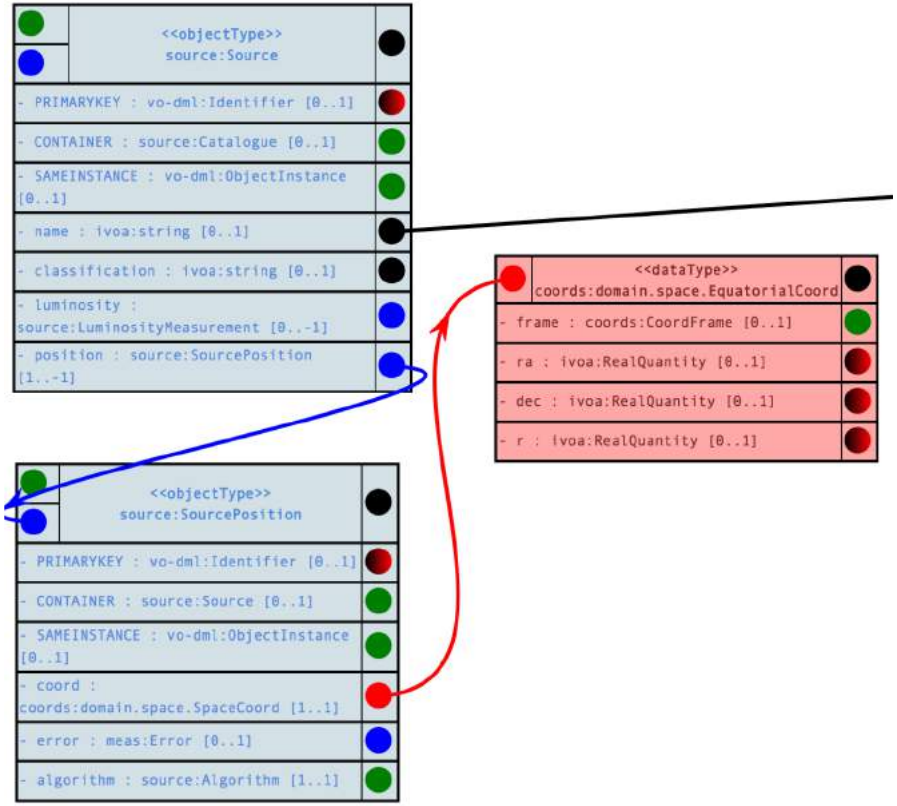
# Source.name is “MatchID” FIELD



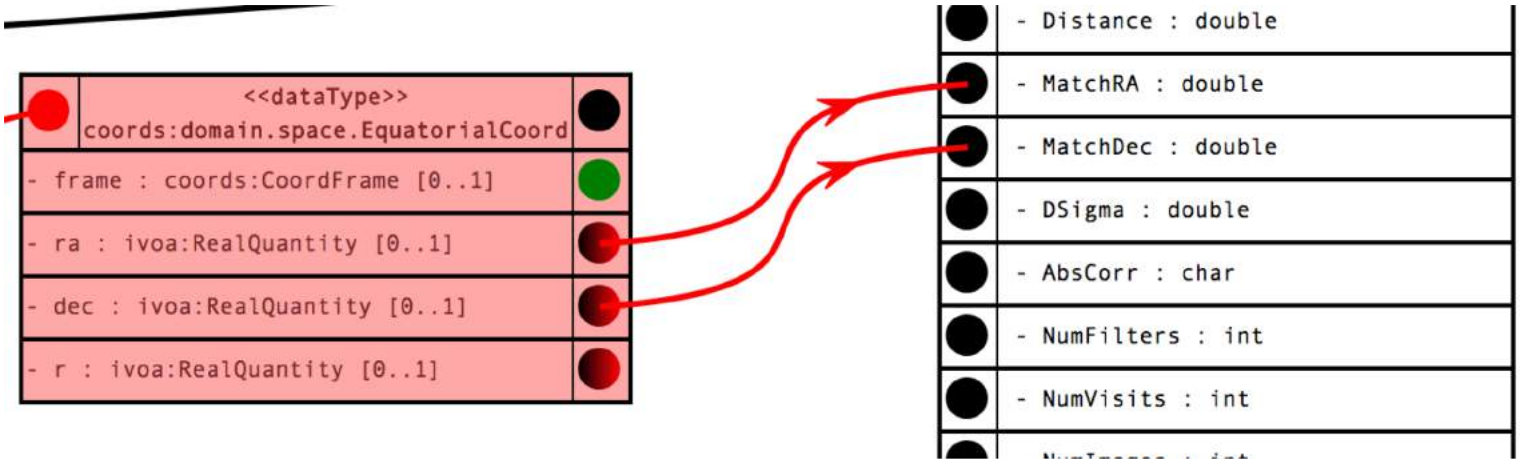
# Source has a SourcePosition



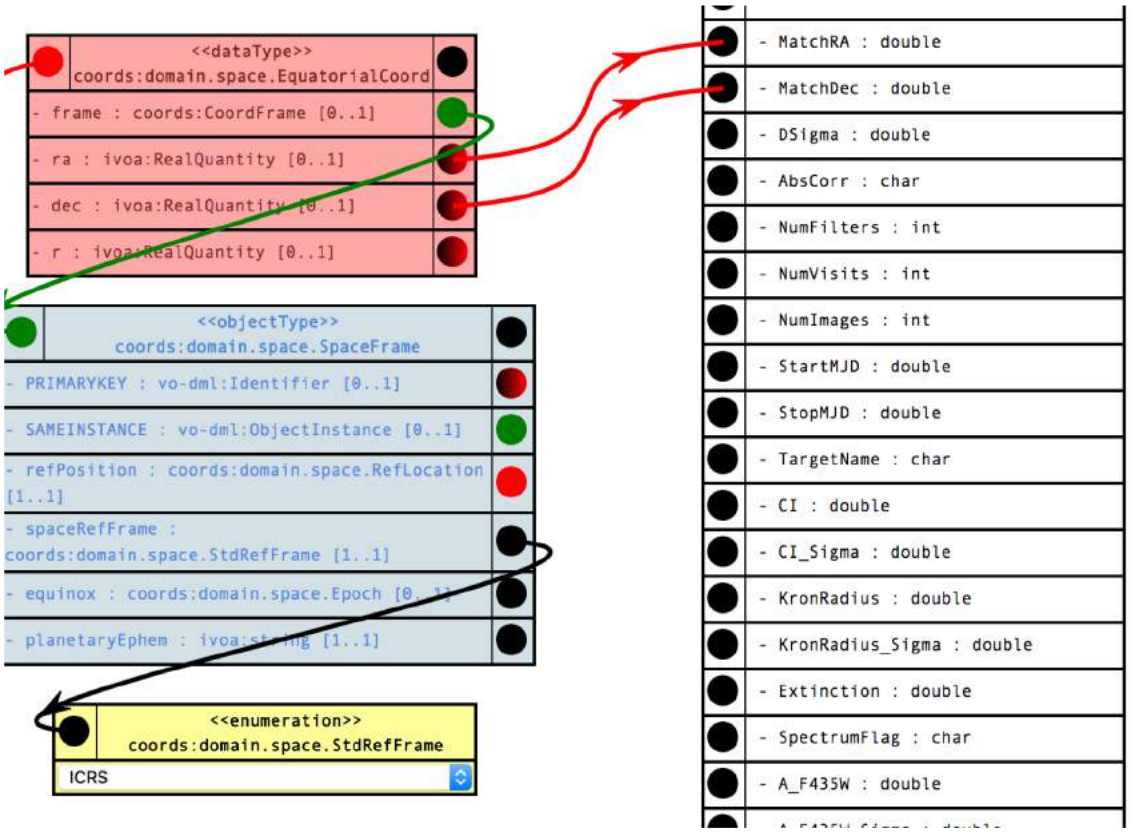
# SourcePosition has an EquatorialCoord



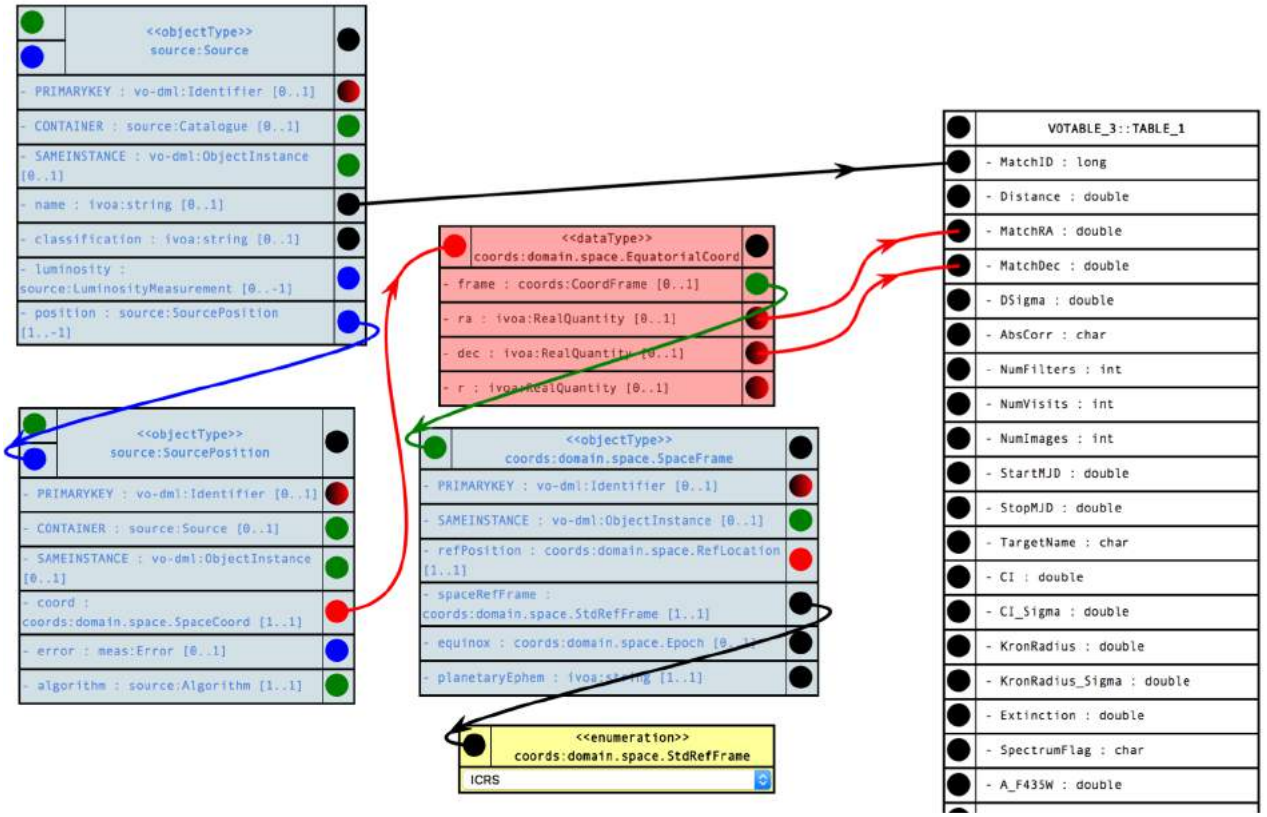
# Map the ra and dec to appropriate FIELDs



# Add the ICRS frame



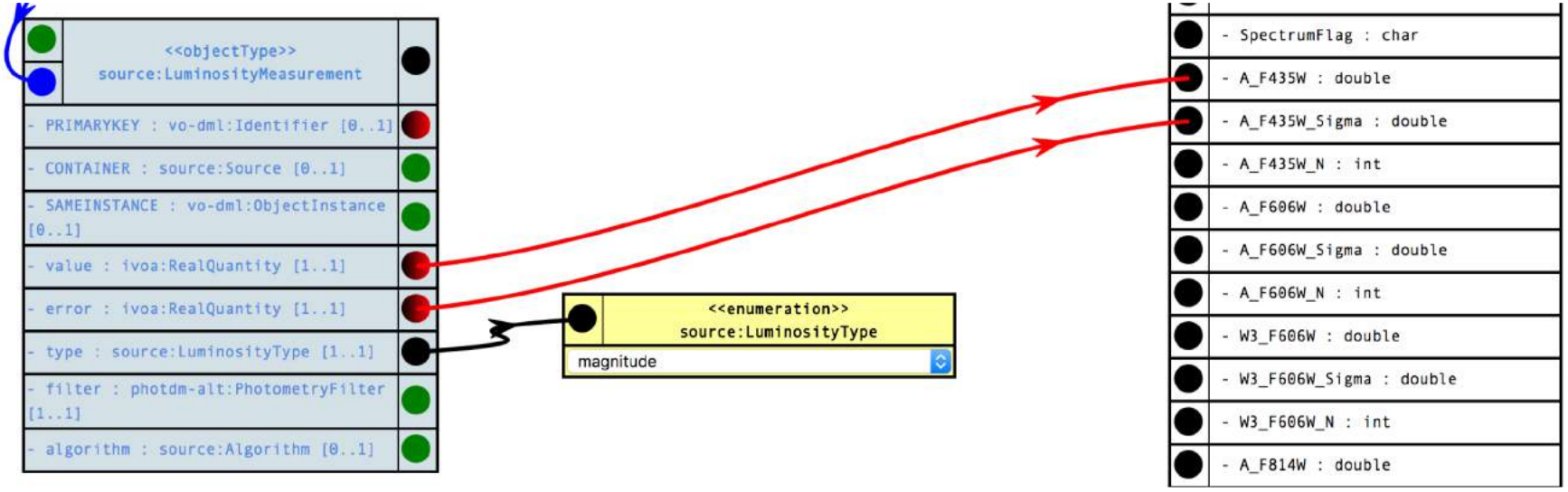
# Complete mapping for name and position



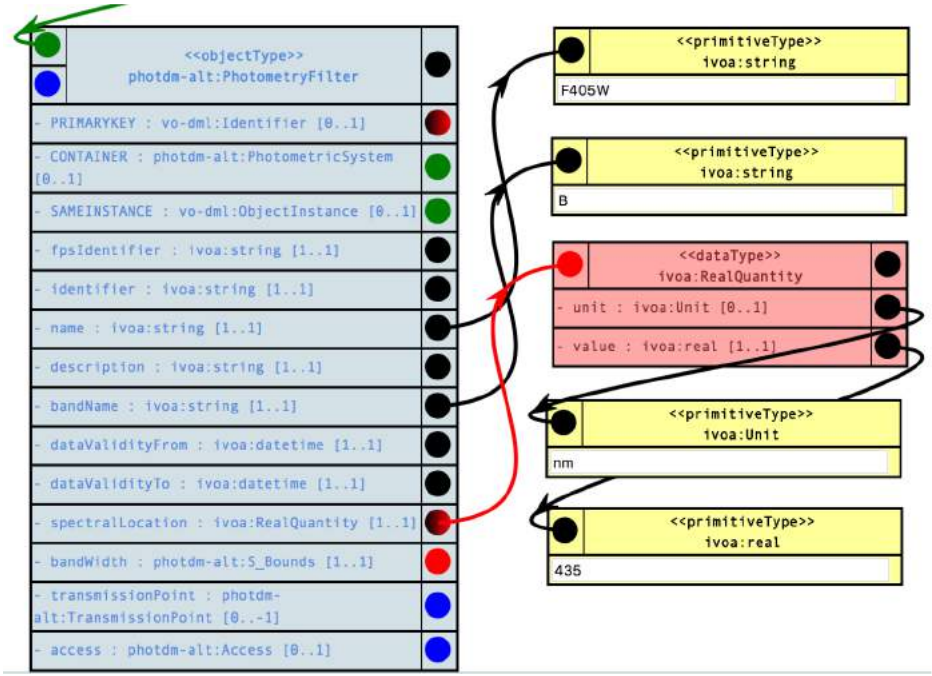
# Complete mapping for name and position - XML

```
<GLOBALS>
    <INSTANCE dmtype="coords:domain.space.SpaceFrame" ID="_icrs_">
        <ATTRIBUTE dmrole="coords:domain.space.SpaceFrame.spaceRefFrame">
            <LITERAL dmtype="coords:domain.space.StdRefFrame" value="ICRS"/>
        </ATTRIBUTE>
    </INSTANCE>
</GLOBALS>
<TEMPLATES tableref="matches">
    <INSTANCE dmtype="source:Source">
        <ATTRIBUTE dmrole="source:Source.name">
            <COLUMN dmtype="ivoa:string" ref="MatchID"/>
        </ATTRIBUTE>
        <COMPOSITION dmrole="source:Source.position">
            <INSTANCE dmtype="source:SourcePosition">
                <ATTRIBUTE dmrole="meas:CoordMeasure.coord">
                    <INSTANCE dmtype="coords:domain.space.EquatorialCoord">
                        <ATTRIBUTE dmrole="coords:domain.space.EquatorialCoord.ra">
                            <COLUMN dmtype="ivoa:RealQuantity" ref="MatchRA"/>
                        </ATTRIBUTE>
                        <ATTRIBUTE dmrole="coords:domain.space.EquatorialCoord.dec">
                            <COLUMN dmtype="ivoa:RealQuantity" ref="MatchDec"/>
                        </ATTRIBUTE>
                        <REFERENCE dmrole="coords:Coordinate.frame">
                            <IDREF>_icrs_</IDREF>
                        </REFERENCE>
                    </INSTANCE>
                </ATTRIBUTE>
            </INSTANCE>
        </COMPOSITION>
    </INSTANCE>
</TEMPLATES>
```

# Map the luminosities

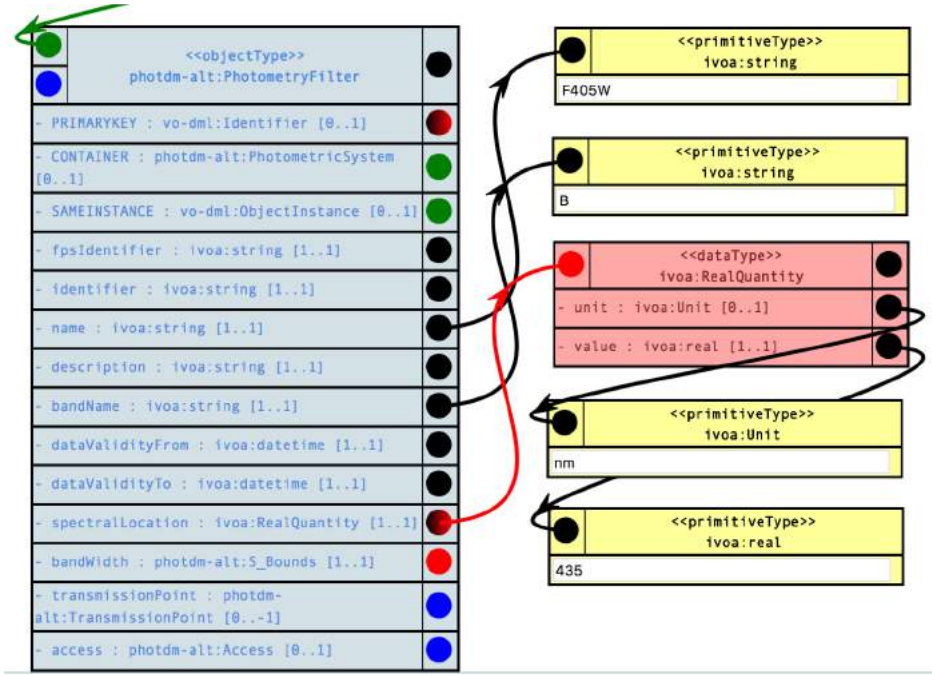


# Define the filters



## Define the filters

- This information was not in the original VOTable.
- By adding these filter instances, the VOTable becomes more informative.




```
<INSTANCE ID="_A_F435W" dmtype="photdm-alt:PhotometryFilter">
   <PRIMARYKEY>
      <PKFIELD>
         <LITERAL value="A_F435W" dmtype="ivoa:string"/>
      </PKFIELD>
   </PRIMARYKEY>
   <ATTRIBUTE dmrole="photdm-alt:PhotometryFilter.name">
      <LITERAL value="A_F435W" dmtype="ivoa:string"/>
   </ATTRIBUTE>
   <ATTRIBUTE dmrole="photdm-alt:PhotometryFilter.bandName">
      <LITERAL value="B" dmtype="ivoa:string"/>
   </ATTRIBUTE>
   <ATTRIBUTE dmrole="photdm-alt:PhotometryFilter.spectralLocation">
      <LITERAL value="435" dmtype="ivoa:RealQuantity"/>
   </ATTRIBUTE>
</INSTANCE>
```

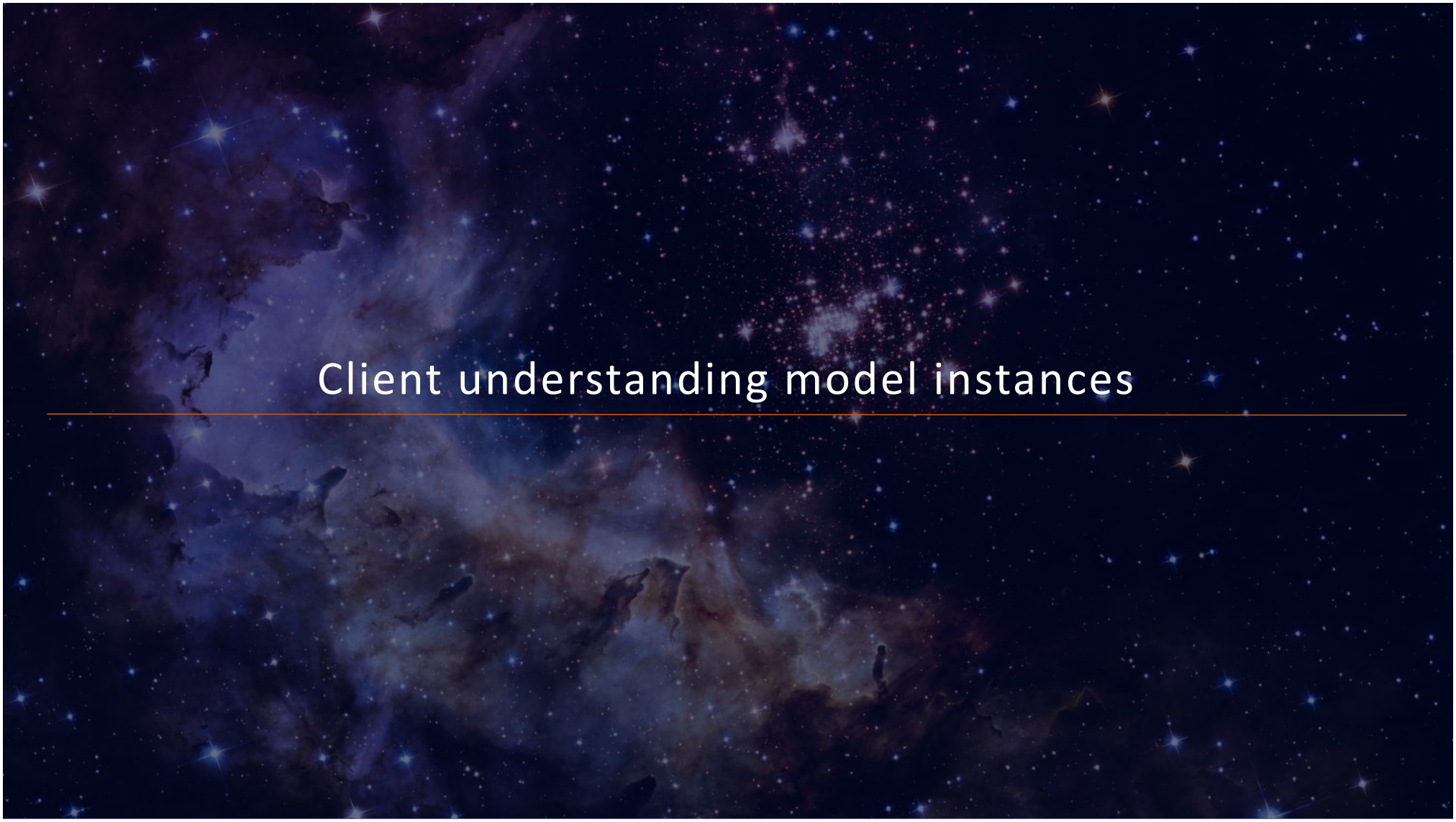
## Add annotations to existing services

Using these mapping techniques, VO-DML annotations were added to 2 MAST services:

- HSC Cone Search annotated with one Source instance per row:
  - https://masttest.stsci.edu/vodml/Mashup/VoQuery.asmx/HscCone?RA=[RA]&DEC=[Dec]&SR=[radius]
  - Each Source is a “match”, i.e., a cross-correlation of one or more source detections.
  - Each source has:
    - A MatchID identifying the match
    - One or more luminosities
- HSC Detection search annotated with one Detection per row:
  - https://masttest.stsci.edu/vodml/Mashup/VoQuery.asmx/HscDetections?MATCHID=[MatchID]
  - Each Detection has exactly one:
    - URL to the image from which the Detection was extracted
    - Luminosity, measured from the Detection’s image

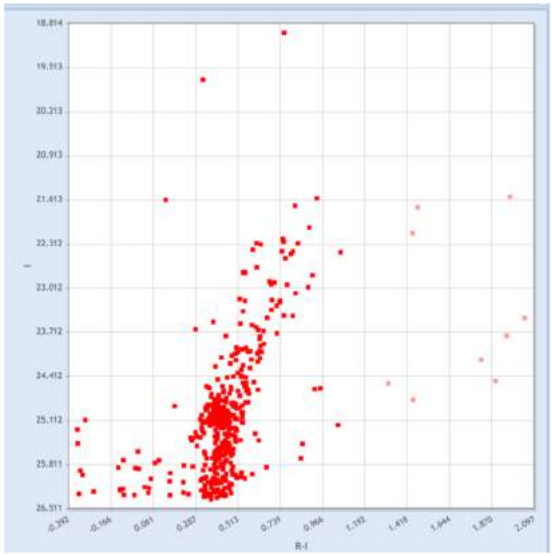
## Data provider feedback

- Creating annotations by hand was error prone for me.
  - Less so when I was more familiar with VO-DML models.
- With Mapper tool, annotation creation was a couple orders of magnitudes faster.
- For any annotations, validators will be very important.
- Existing VOTables don't always map cleanly to standard models.
  - Sometimes it may be worth tweaking the existing service to make the output fit the model a little better.
- Referencing global instances (like photometry filters) by values from the table data is supported, but details not obvious.


Client understanding model instances

# MAST Portal recognizes the VO-DML objects

- __dmtype__: source:Source
- __id__: _source
- name: 17112744
- luminosity
  - 0
    - __dmtype__: source:LuminosityMeasurement
    - value
    - error
    - type: magnitude
    - filter: _W2_F336W
  - 1
    - __dmtype__: source:LuminosityMeasurement
    - value
    - error
    - type: magnitude
    - filter: _W2_F439W
  - 2
    - __dmtype__: source:LuminosityMeasurement
    - value: 19.6096000671387
    - error
    - type: magnitude
    - filter: _W2_F502N
  - 3
    - __dmtype__: source:LuminosityMeasurement
    - value: 19.1471004486084
    - error
    - type: magnitude
    - filter: _W2_F547M



- __dmtype__: photdm-alt:PhotometryFilter
- __id__: _W2_F502N
- __primarykey__: W2_F502N
- __attributes__
  - name: W2_F502N
  - bandName: V
  - spectralLocation: 502

## Using Python to follow example workflow

- See notebook: https://github.com/olaurino/ivoa-dm-examples/blob/master/notebooks/HSC-Example.ipynb
- Perform a cone search which returns a VO-DML aware result.

```
def cone_search(base_url, coord: SkyCoord, radius):
    # Make sure the coordinates are expressed in ICRS and degrees
    ra = coord.icrs.ra.deg
    dec = coord.icrs.dec.deg
    # Make sure radius is in degrees
    radius_deg = radius.to(u.deg).value

    # Cone search params.
    params = {'RA': ra, 'DEC': dec, 'SR': radius_deg}

    return read_url(base_url, params)

# Define Cone Search parameters
search_coord = SkyCoord.from_name("m82")
search_radius = 0.001 * u.deg

# HSC Cone Base URL (in general could come from registry)
HSC_MATCHES_URL = 'https://masttest.stsci.edu/vodml/Mashup/VoQuery.asmx/HscCone?'

# Perform search.  Result is VO-DML aware parser object.
hsc_matches_results = cone_search(HSC_MATCHES_URL, search_coord, search_radius)
```

# Classic astropy table results are available.

```
# Get Astropy table representing the results
# kind of a hack, we need to improve the API
matches_table = list(hsc_matches_results.tables.values())[0]

# Show info
print(f'{source_template.cardinality} matches found.')
matches_table.show_in_notebook()
```

```
19 matches found.
```

*QTable masked=True length=19*

Show 50 entries Search:

| idx | MatchID | Distance | MatchRA | MatchDec | DSigma | AbsCorr | NumFilters | NumVisits | NumImages | StartMJD | StopMJD |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | none | arcminutes | deg | deg | mas | | | | | days | days |
| 0 | 17350615.0 | 0.00883306720656974 | 148.96888203758 | 69.6797072526489 | 0.0 | Y | 1 | 1 | 1 | 54834.6258381181 | 54834.6575510763 |
| 1 | 17112744.0 | 0.0131180875647911 | 148.968997760412 | 69.6795900802937 | 24.9352755222915 | Y | 5 | 2 | 7 | 50523.2968749586 | 51053.1723864423 |
| 2 | 17115228.0 | 0.020600919203083 | 148.968682222876 | 69.6800372272498 | 0.0 | N | 2 | 1 | 2 | 50706.0557152592 | 50706.5385857429 |
| 3 | 17111544.0 | 0.0303987662778864 | 148.967404818518 | 69.6793523018495 | 0.0 | Y | 1 | 1 | 1 | 51053.0043308656 | 51053.0177567475 |
| 4 | 17109683.0 | 0.0319055670213261 | 148.968358648006 | 69.6802334322381 | 0.0 | N | 3 | 1 | 3 | 50706.0557152592 | 50706.5385857429 |
| 5 | 17114183.0 | 0.032588602427965 | 148.96711764508 | 69.6794230683398 | 0.0 | Y | 2 | 1 | 2 | 51053.0043308656 | 51053.0362752476 |

# Extract the Source instances

```
# By default tabular instances are described as "templates" that provide
# access to individual columns. We'll assume there is only one Source template per row.
source_template = hsc_matches_results.find_instances(Source)[0]

# We can unroll the templates to get a list of Sources, one per row.
sources = unroll(source_template)
```

# Demonstrate accessing Source attributes

```
def print_source_properties(source):
    name = int(source.name.value)
    sky_coord = source.position[0].coord
    print(f'\nSource {name}\nposition: {sky_coord}''')

    for luminosity in source.luminosity:
        band_name = luminosity.filter.band_name
        mag = luminosity.value.value if luminosity.type == 'magnitude' else np.nan
        if not np.isnan(mag):
            print (f'    {band_name} mag: {mag},  ')

for source in sources:
    print_source_properties(source)
```

```
Source 17111544
position: <SkyCoord (ICRS): (ra, dec) in deg
    (148.96740482, 69.6793523)>
    R mag: 19.2964992523193,

Source 17109683
position: <SkyCoord (ICRS): (ra, dec) in deg
    (148.96835865, 69.68023343)>
    B mag: 22.9946994781494,
    V mag: 20.3750991821289,
    I mag: 18.8915004730225,
```

## Using source attributes, various selections can be done

- Of sources with multiple luminosities, select the one with the brightest B magnitude.

```
def find_brightest(sources, band_name):
    [snip...]

brightest = find_brightest(sources, "B")
print(int(brightest.name.value))
```

```
17109683
```

# Follow-up query to find all Detections for selected Source

- Follow-up query may be discoverable via Provenance or DataLink techniques?

```
HSC_DETECTIONS_BASE_URL = 'https://masttest.stsci.edu/vodml/Mashup/VoQuery.asmx/HscDetections?'
params = {'MATCHID': brightest.name.value}
detection_results = read_url(HSC_DETECTIONS_BASE_URL, params)

detection_template = detection_results.find_instances(Detection)[0]
print(f"{detection_template.cardinality} detections found")
detections = unroll(detection_template)

detections_table = list(detection_results.tables.values())[0]
detections_table.show_in_notebook()
```

```
3 detections found
```

*QTable masked=True length=3*

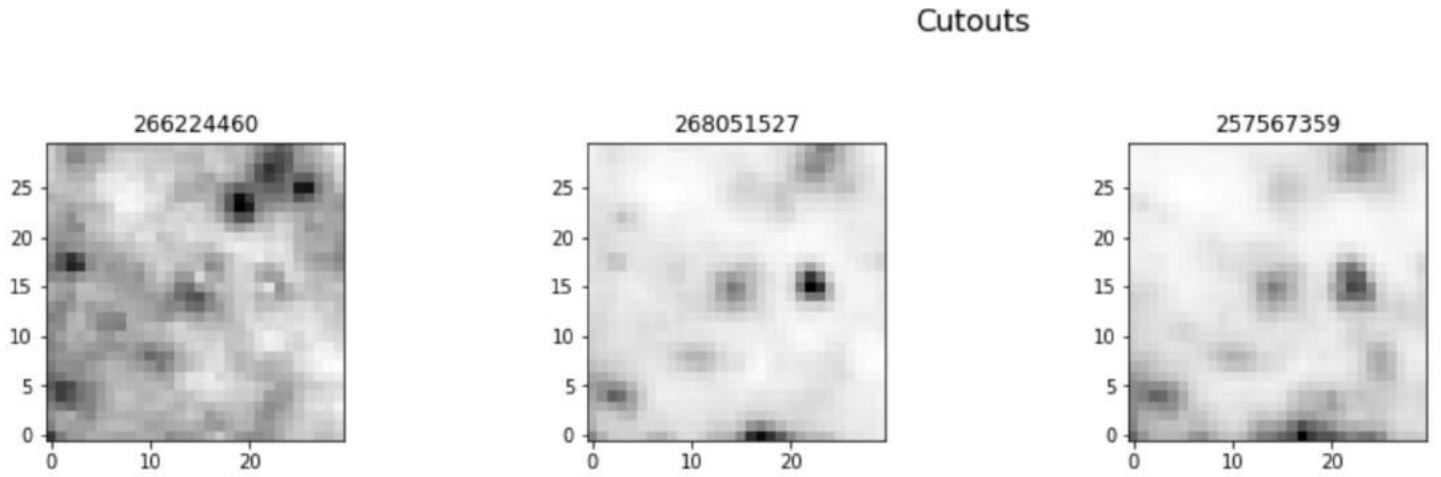
Show 50 entries Search:

| idx | CatID | MatchID | MemID | SourceID | ImageID | Det | MatchRA | MatchDec | SourceRA | SourceDec |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | none | none | | deg | deg | deg | deg |
| 0 | 266224460 | 17109683 | 1 | 4662322165.0 | 550998.0 | Y | 148.968358648006 | 69.6802334322381 | 148.968358648006 | 69.6802334322381 |

# Access Detection images and plot cutouts around the Source

```
for idx, detection in enumerate(detections):
    #Make the title be the name of the detection.
    title = detection.name
    ra = detection.position[0].coord.icrs.ra.deg
    dec = detection.position[0].coord.icrs.dec.deg

    # Plot the first image data we find.
    filename = detection.source_image[0].url
    download_and_plot_cutout(title, ra, dec, filename)
```



## Client development feedback

- Understanding VO-DML concepts was important for writing an effective parser.
- While the annotation syntax is large, it just reflects the structure of VO-DML itself.
  - General purpose clients need to understand those concepts.
  - With those concepts understood, the (rather large) syntax is easily parseable.
  - Clients focused on a particular data model or use case may not care about all the details of VO-DML concepts.
- Libraries like rama will help for creating real instances of any model.
  - My C# parser was derived from a parser that handled the original syntax, so its design was very complex.
- Need to clarify desired behavior mapping between VOTable primitive types and VO-DML primitive types.
- This (rama) design makes it clear that code handling VO-DML can be kept separate from the rest of VOTable parsing.

# Thank you!