# IVOA Nov 2023 DAL 1

Time: Fri 10 Nov 14:00 MST **[session #3]**

## ConeSearch 1.1

- Previous 1.1 tried attempts to do too much
- What's planned:
    - UCD1+
    - some DALI features: MAXREC, RESPONSEFORMAT, ...

- update to UCD1+ (issue #20)
- re-add MAXREC – re-add RESPONSEFORMAT
- rework resource description (a.k.a. Section 3) – (TBD – issue #54)
- align VOTable error to DALI – (TBD – issue #17)
- fix silently failing optional parameters

4 open PRs currently

- Reuse of text from previous 1.1 work so shouldn;t be too controvertial

Discussion (that can fit in v1.1)

- main ID usage (GitHub issue #53)
- Handling unrecognised (custom/optional) parameters (GitHub issue #17)
    - Currently (v1.0) unknown parameters are just ignored
    - Have to recheck discussion - but maybe an INOF element that records the ignored parameters

Other fixes/updates that can fit in v1.1

- OVERFLOW approach
- https
- VOTable examples
- Update .xds and .vor

Major (Postponed to 2.0)

- Query param for catalogue selection
- Allow ST-MOC query - desirable but not a minor change
- Source DM annotation (maybe)
- Multi cone
- Discovery of versioned CS services

Fixes/updates

- Align with SIA/SODA/DALI
- Adding support for POST - but major deprecation required to support this and it made a mess of the text.

Should be major or minor?

- EPOCH query param (transform data to that epoch)
- TIME parameter as a filter
- Max time span allowed
- all sky search (e.g. Time only filtering)
- Add time interval
- Discoverability of services supporting time

Not in the current issue list

- suggest 1.x
    - Results ORDERing
    - ConeSearch as a TAP profile
- Suggest 2.x
    - /capabilities
    - DALI compliance

Timeline: WD by Sydney (PR pending validators and implementations)

Suggestion for 1.2 - add an IVO id that could be used in the query

## Datalink service descriptors in Firefly

Describe in detail how we are using

- Datalink
- Service descriptors

Point out we are solving problems

Search results

- are straight forward
- Using local_semantics (a lot) and content_qualifier (just starting)
- example - 4 images for #this - grid plus RGB bandds - show as 4 images on results
- spectra - #spectrum-grid, #spectrum-combined
- Click on 1 row, show two spectra

Search UI

- challenging as it needs to be based on the service descriptor
- Metadata driven
- Need to create a UI for the service
- May result in multipel searches when a user clciks search

Data collection explorer

- List of datasets
- Search area changes depending on what dataset is selected

Process:

- DCE requests an Obscore like table to list datasets
- Each dataset return datalink table for list of concurrent searches for a dataset
- Primary SD describes both UI and search

Problem 1 - how searches relate

- Using datalink to define searches (not supplying results)
- Need #primary-query (e.g. SIA2, SSA), #concurrent-query (e.g. SIA2, TAP)
- Then run concurrent searches and translate inout between service descriptors

Problem 2: Search user interface

- Presenttion of HiPS (e.g. FOV, color, center)
- Astronomers want examples that match service and field plus link to describe dataset
- Enhanced group (ui group) in service descriptor

Problem 3: TAP Service Descriptors

- Want to use cataloges in search results -> TAP
- Must be solved in 2024
- Mark it as a TAP service
- Inpout params has placeholders
- Add tokenSub group to define template parameters
- Build UI based on template params
- Thus need templating in service descriptors (e.g. 'SELECT ... FROM ... WHERE ... POINT(${ra}, ${dec}) ...')

Problem 3a: TAP Upload

- Service descriptors are messy
- Need to provide parameters for uplaod tables
- (table name,  ra and dec columns in upload table)
- How do we specify service descriptor for uploads?
- Need some good idea on how to do this

Rabbit Trail

- How to do help for TAP table
- Users want to click link to HTML display of info/help about table and/or columns
- Needed to replace existing service with TAP

Successes

- Datalink working an getting better
- Datalink one-to-many relationship is solves a 15 year problem at IRSA
- Service descriptors are huge break through for searching
- Service descriptors separate the UI from the search

Suggestions

- Standard way to extend service descriptors
    - Named extensions could be recognized
    - *For example*- Rubin and IRSA might use same extension
- Templating service descriptors
- Cross link HTML documentation (i.e. between specs)

JD: TAP example might be a path to get html help
FB: Templating has been proposed but would be great to have an example implementation

MT: Document could be generated by querying TAP serfvice for table metadata and then use that to build UI
GDF: We already do that, but doesn't have the capability to provide really rich doco. Trey working on a datalink solution for this which he will discuss at the next interop

GDF: In the templating proposed so far there is no way to construct restful interfaces as you can't include variables in path

## Object Observability Simple Access Protocol status

Driver is need for coordinated observations between observatories, particularly in age of GW/multi-messenger astronomy and time domain astronomy

Aim t*o standaridize the retrieval of the well-known observability or contraint-free periods of astronomical targets that all facilities have*

ObsObSAP has been around since 2020, has an existing client and the service is implementated at numerous observatories

Existing tools reviewed

Next steps: Object Observability (or Target Observability)

CA: Maybe avoid TO or ToO as that is assumed to be Target of Opportunity
CA: Can still change name in 1.0 as it hasn't been endorsed. v1.1 implies 1.0 already released

# Discussion

## Modernisation of protocols - improve compatability with modern web dev tools

GDF: Still very interested in modern definition of protocols/APIs (e..g JSON/ OpenAPI) to assist service implementors to use modern tools. High priority for Rubin
FB: How would that look like
GDF: Way more services than clients - clients have resources to stay up to date - would need to update clients to support new services while supporting old services. New services would be json based
GDF: Suggest a tiger team as no one service can do this alone
GM: Have used microsservices with json to send votable in the new SIMBAD's ConeSearch service
GDF: About service interaction not votable format change - so how UWS works. Really hard to implement UWS in contemporary
JD: Is OpenAPI a useful step?
Tim: Is there support for json? Yes, then presumably some would be intersted in volunteering.

RE: NED already has json behind the scenes and easier to convert json

DM: Suggest updating standards as defining the data model and then have an examples of the json implementation as well as XML. Don't want to be talking in 10 years about how to convert away from json!

Joshua Fraustro: MAST has a model of the UWS interactions - need management approval to make them open source and share.
Tim: Use Pydantic and FastAPI too - very happy with them

**Volunteers for the tiger team:**

- Dave Morris <dave@metagrid.co.uk>
- If we are volunteering to help with creating service APIs in OpenAPI, I would participate: kgillies@tmt.org

- dower@stsci.edu - work on open sourcing the pydantic model code in recent MAST TAP services.
- jfraustro@stsci.edu sharing pydantic/pydantic-xml models
- Rick Ebert <rick.ebert@caltech.edu>  (NED)  anything that helps to standardize datamodel/semantics
- James Dempsey <james.dempsey@csiro.au
- Grégory Mantelet <gregory.mantelet@astro.unistra.fr>
- Marco Molinaro <marco.molinaro@inaf.it>
- Paul Harrison <paul.harrison@manchester.ac.uk>