

DAL session - IVOA Interop. - Nov. 2025

Date: 14th November 2025 - 16:00

Implementing Datalink in VizieR (*Ivan Brossard*)

- "VP" links on VizieR catalog page display velocity plots; Simbad links open Simbad (on the HTML website)
- These are not accessible with -- any non-html output
- Frequent issue: 4000+ VizieR catalogs have one linked data
 - Graphs and other displays are calculated on the fly
- Solution: DataLink which table contains ID and access_url
 - ID is used as the identifier for the 'thing you want to know more about'
- Example for a SCS query : the result VOTable with a table having a _ivoid column
- New way: `<LINK>` => DataLink table URL. That table contains: ivoid, description, and access_url which links to the data to plot
- Example: from a VizieR catalog page, there is a nice plot only visible on the website
- Access points
 - can use TOPCAT which has a nice "Invoke Service" Activation Action. Then click on main table row to plot the data
 - astropy's getdatalink()
 - in Aladin one can click on the table at the bottom of the screen
- Next: currently DataLink is not completely implemented for all VizieR catalogs
- A new section in the "Download Notebook" is being added to download a pre-filled Notebook which demonstrates how to run for datalink example usage
- Still need to add FITS images
- Francois Bonnarel: Has it been tested with Python tools?
 - Yes, (pyvo/astropy)

Creating a tool to automatically generate an ObsCore Table from a FITS collection (*Nathan Barloy*)

- Context: VizieR catalogs indexing with an ObsCore database called associated data
- ObsCore data is ingested with Saada - <http://saada.unistra.fr/saada/>
 - Only ~3% of the catalogs are indexed in assocdata database
 - Adding more data in tables is difficult with Saada
 - Solution: new Python library - "ObsCore Builder"
- CDS-ODAS: Open Data for Astronomy: make data more open and findable

- Objectives: a simple and easy to use internal tool to ingest associated data with a Python Library, to allow for access of assoc data via TAP, SIA, SSA...
- Create a new web page for catalog submissions with a new workflow to validate the ObsCore table generated
- How does it work ? => Steps for ObsCOrE generation
 - automatic extraction from FITS file
 - based on Astropy to read the FITS headers, common keywords + WCS solved with Astropy
 - Need to be flexible ex: with non-std values such as CTYPE1='LINEAR' in spectra FITS files ! Sometimes inference could be used but shouldn't ex CTYPE1='WAVE' but no CUNIT1 given, the unit can be given in the comments. In general non-std rules are accepted
 - Data Curation with ext services
 - Fetch missing info from Simbad if obj name given
 - If wavebands are missing, get them from the Observatory/Instrument/Filter using a local database of filters created at CDS
 - Manual addition to the table from the people who submit the data
 - Problem: The FITS files received cannot be modified, and some catalogs share a common format
 - Solution: A Mapping file (using Python code) has been developed, allowing the user to provide additional info. This file can be provided when extracting an ObsCore
 - Principle: for each line it's possible to add a default value for a FITS kw, force a value in a ObsCore field, fill an obscore field, reference some env vars etc.
 - Example: Python usage of this Mapping file: `extract(catalog_dir, mapping_file)`. The mapping can also be applied after the obscore_table has been generated
 - Next ?
 - support more non-std FITS formats
 - a CLI tool easily installable by the CDS staff to test the library
 - a new webpage for data submission and workflow to use it
 - Open source the library with Pypi
 - Conclusion
 - An easier ingestion process for both the CDS staff and users, which makes data more Findable
 - Library still in development, things can change
- Questions:
 - James Tocknell: How scalable is the library (re: corner cases) vs having a directory of configs
 - Not intended to cover every case-- sometimes the current workflow can't find information, can be improved
 - Marion Marchand: how do you handle the security for the code embedded in the mapping file ?

- could be restricted to some functions
- No threat because the Python library runs on the user's computer (for the case when the python library is used locally or for the cli tool)
- Cannot create users to make Mapping files and run at CDS: in this case only a subset of functions will be available for the ingestion process (for the case when this is the hosted webpage submission process: no custom python function available)

ESDC TAP Stateless Follow-up (*Jose Osinde Lopez*)

- Follow-up to June 2025 UMD presentation.
- Jose maintains TAP library at ESAC
- Goal: make the library...
 - stateless
 - Horizontal Scalability - spin-up / remove instances based on load
 - instances are independant
 - High Availability / Load balancing
 - Requests can be routed to any instance
 - If one fails, another continues processing
 - Easier to deploy
- Problems:
 - TAP is not intrinsically stateless
 - Events
 - Job status, quota updates, logs
 - User status
 - uploads, format conversions
 - Prevents a true stateless layout
- Approaches:
 - Application state stored in memory between instances and synchronize:
 - => syncing info between instances is too difficult
 - => refactor the existing TAP service and isolate the state outside the instance
 - But where to put the state?
 - Archive database is first option
 - Pros:
 - Already a central structured information store
 - the DB guarantee data persistance across restarts, deployments and crashes
 - All instances read/write to same database
 - Intrinsically atomic operations across concurrent access, transactions

- Avoids adding systems
- Cons:
 - In some cases the TAP service couldn't safely update the state
 - Databases were locked in other transactions
- ESDC uses Postgres
 - good for OLTP - small frequent parallel transactions
- New archives with big data (GAIA, Euclid...) are using OLAP
 - OLAP - analytical, huge queries
 - The cause of transaction-locked state update failures
- Problems:
 - Some processes generated multiple parallel events
 - database cannot handle them efficiently-- concurrency conflicts and performance bottlenecks
 - Matching in-memory speed is difficult, important for critical operations
- Alternative: in-memory DBs are a better
 - Use cases:
 - Caching, session storage
 - Advantage
 - Low latency, high speed, concurrent and parallel access
 - reduces load on traditional DB -- very important
 - Cons:
 - Data is intrinsically volatile until/unless it is persisted somewhere
 - careful consideration needed of what is in-memory vs persisted in DB
 - If one instance fails, the state could become no longer synchronized
 - Adds an additional point of failure
 - Management overhead - monitoring, logging, upgrades, etc.
 - Popular services like Redis, Memcached, Apache Ignite have to be evaluated.
 - Redis - key/value store with persistence, pub/sub and advanced data structures
 - Memcached - simple caching for temp data
 - Hazelcast/Apache Ignite - in mem data grids for distributed apps
- Strategies:
 - One instance per Archive
 - Failure isolations
 - One instance for ESDC
 - less management
 - Critical failure component
 - similar approaches exist for other services, so it is perhaps tolerable

- Testing = a critical part of the sw, need to be included as part of the validation process, especially stress tests
 - use realistic testing environments (same database versions...)
 - know the limits and expected load to make sure perf degradation stay within
 - Your system may pass unit tests, but not with a large number of users in parallel!
 - May not be known until 1. operational deployment or 2. proper stress testing

Q&As

- ?: Best in-memory DB option ? - not chosen yet
- GM To all: what solution have you chosen ?
 - James Tocknell: have used redis, but for things like SIA

ConeSearch-1.1 status (*Marco Molinaro*)

- 1st attempt was in 2020 and tries to change a lot of things
- Nowadays, the goal is first to be DALI compliant (e.g. adding RESPONSEFORMAT) and to fix UCDs (using now UCD1+)
- At Astro-CC forum:
 - reopened the issue for behavior in reponses to opt params
 - SCS—DALI—> DataOrigin ref chain
 - discuss meta.id; meta.main col unique behaviour in the model
- TODO:
 - no need to discuss: OpenAPI, ivoid in VOTable (DataOrigin), .xsd should be replaced by OpenAPI, ...
 - Clarification of MAXREC vs SR queries for metadata
 - editorial work
 - VOTable media-type
 - ..
- Finally: let's have WD out, when out, check for ref implementations, check for validators
- Discussion issues:
 - DataOrigin -
 - when can an EN be referenced?
 - Wider scope than SCS, reference through DALI
 - media-type
 - Should follow VOTable REC -- application/x-votable+xml
 - Accept txt/xml;votable only for back-compat
 - No disagreement in the attendance
 - Non-votable responses

- Left to DALI, unspecified in SCS
 - Not in SCS's purview to describe it
 - API description
 - move from XSD to OpenAPI
 - No disagreement in the attendance
- Optional / Custom Parameters: GitHub issue #17
 - Avoid silently failing to recognize non-mandatory parameters
 - Don't **force** clients to parse a complicated warning message
 - But also provide potentially useful information for users/clients
 - Use `<INFO name="warning" value="Unrecognized optional/custom parameters."/>`
 - GM: Already implemented in the SIMBAD's ConeSearch service
- Metadata Queries:
 - SR=0 in v1.03
 - Originally: *might *be used for metadata responses
 - Not binding requirement!
 - v1.1 MAXREC=0
 - Current DALI method for metadata
 - But a service will fail on MAXREC only requests because SR is mandatory!
 - Message to validator maintainers:
 - SR=0 and data response is not an error
 - No way to address behavior in other ways without breaking change (v2.0)
- Main ID - GitHub issue #53
 - Issue text:
 - "(meta.id;meta.main) may not be repeated in the table..."
 - meta.id;meta.main must be set only on a single result FIELD
 - but, should it be unique in the catalogue?
 - Gille Landais: consider timeseries data: same object, position, observations at different times, ergo different ID's
 - Spectra data: one source, one position, multiple spectra
 - Cone search can still be used to search for
 - Suggest changing from 'may not be repeated' to 'should not be repeated'
 - James Tocknell: Was it a "may" because multiple catalogs may have been stacked and not checked?
 - "may" should be a warning to validators
 - Pat Dowler: 'may' probably meant to be used as "MUST" in actuality-- "MUST not be repeated"
 - ?: Suggestion remove the negative 'not' and allow it to be a permissive 'may'

