Rubin's use of IVOA standards - An update

Gregory Dubois-Felsmann, Caltech/IPAC
RSP Product Owner

11 May 2023

The Rubin Science Platform *Aspects*

- Portal Aspect
  - Implemented with IPAC Firefly, an open-source astronomical data access and visualization toolkit
  - Almost entirely based on either "1st class" IVOA query protocols or lower-level services via DataLink
  - Really a general IVOA data explorer - only superficial Rubin-specifics (skinning, help text, defaults…)
- Notebook Aspect
  - JupyterLab, JupyterHub, Rubin-created deployment & launch tooling
  - Data services are either accessed from our (and external) IVOA services via PyVO, or for file-like data via the Rubin Butler Python API
- API Aspect
  - "1st class" IVOA protocol standards (TAP, ObsCore, DataLink, SODA, …)
  - Rubin-specific services based on DALI, VOSI, etc. with service parameters discovered via DataLink service descriptors
    - "Heavyweight" services for forced photometry, data product re-creation, bulk cutouts
    - *Many* microservices supporting links between datasets (time series, coadd inputs, bulk cutouts)

Where are we?

- We have TAP over the Rubin open-source *Qserv* distributed spatially shared DB
  - Based on OpenCADC TAP, with extensions for DataLink (to be upstreamed)
  - Also supports ObsTAP for the static image/file data products in data releases
- We use the “links service” model for data access from ObsCore
  - Links service implemented as FastAPI microservice
- Basic SODA cutout service implemented (see Frossie Economou / Russ Albert talk)
- We have a few basic query microservices, discoverable from catalogs via DataLink, including time-series retrieval for a designated object (calls through to TAP)

- We implemented HiPS and MOC generation from scratch in the Rubin framework
  - Optimal resampling to HEALPix – full-resolution science-grade HiPS tiles
  - Open-source
  - Integrated into the Rubin pipeline production environment
  - Hue-preserving color

What are we still working on? (1)

- Add temporary-table upload support to Qserv for multi-object searches
  - Also add to TAP (CADC TAP supports DALI UPLOAD, but we need to code the interface to Qserv)
- Add persistent user-database support
  - Planning to use CADC “YouCat” extensions to TAP
- Support access to data documentation in a fully interoperable, fine-grained (down to column-level) manner

- Support of our dynamic (nightly) “Prompt” data products (images, diffim/transient catalogs, Solar System data)
  - Planning for this all to be on Postgres
  - Same richness of DataLink services required

  - Nightly MOCs (x6 filters) or nightly-updated STMOCs

What are we still working on? (2)

- Many more microservices
- Forced-photometry-on-demand service
  - Needs a protocol based on DALI UPLOAD – no well-known existing model for this
- High-multiplicity bulk-cutout service (supporting lightcurves)
  - Hundreds to thousands of science-grade cutouts per call – huge MEFs with several HDUs per cutout? packages of many files?
  - Cubes are not a great solution as there are then complex WCS issues (dither, moving objects)
  - Common problem with other upcoming projects (e.g., NASA SPHEREx)
- Other heavyweight data product creation/re-creation services

What are we still working on? (3)

- ObsLocTAP well-integrated with other services (2-hour advance notice)
- Registry interfaces
  - RegTAP 1.2 support in CADC TAP would be very helpful

Acronyms & Glossary

Stretch goals

- Return “Parquet with VOTable metadata” from query services
- PyVO and UI (Firefly, TOPCAT) support for the discovery of a means to make VODML/MIVOT data models, DataLink service descriptors, and GROUPs (possibly obsoleted by MIVOT) without/before having to execute an actual query
  - Enables self-documentation and UI support for constructing queries that take advantage of all these value-added features
  - IRSA TAP has a model for this that we may explore using in CADC/Rubin TAP

What do we want/need (implementations)

- Integration of our work on CADC TAP for Qserv support and enhanced Datalink support into the upstream

- A full-featured PostgreSQL + PostGIS TAP service would be very useful to us as it would enable deploying to commercial hostless Postgres-as-a-service (where pgsphere is not available)
  - This has a long-standing corridor-talk reputation as hard/impossible but we are not sure why
  - We think this would be an enormous community service in this era of migrating archives to the cloud

What do we want/need (standards - 1)

- Community standardization of a Parquet-based all-sky bulk data access scheme (see Mario Juric talk earlier today)
  - Tiling definition (incl. overlaps)
  - Cloud-friendly data access model
  - Preservation of the same rich metadata (incl. DataLink) from our TAP service
  - ObsCore convention for documenting the individual tiles in such a dataset (by HEALPix)
- VOTable-as-Parquet
  - Notional implementation: devise a way to package all the existing VOTable metadata (units, UCDs, utypes, groups, service descriptors, MIVOT) ***verbatim*** (yes, in XML) as a blob in an IVOA-standardized place in a Parquet file. Advantage: easiest way to leverage existing code in client software; allows client developers to focus on learning to work efficiently with the bulk data in Parquet.

What do we want/need (standards - 2)

- Standardize CADC YouCat for TAP (or some evolution of it)
- Means to make VODML/MIVOT data models, DataLink service descriptors, and GROUPs (possibly obsoleted by MIVOT) accessible from a TAP service without/before having to execute an actual query (c.f. /tables and TAP_SCHEMA)
- Qserv geometry has some limitations w.r.t. the ideal POLYGON standard which we can't currently express in /capabilities (e.g., polygons must be convex).
    - Perhaps we should survey all current mainstream TAPs to see what geometry is really supported and consider adding some more "flags" if there are common threads?
- Would love a clear prescription for how to represent HiPS and MOC datasets in ObsCore – would prefer a unified search interface rather than separating these out