

# TAP: What's missing?

Implementation feedback from TOPCAT's TAP client

Mark Taylor (Bristol)

IVOA Interop  
Sydney

30 October 2015

`$Id: tap-feedback.tex,v 1.7 2015/10/29 22:08:44 mbt Exp $`

# Outline

- List TOPCAT recent improvements
- Review TAP (& related standards) progress since TAP 1.0
- Point up items still requiring **action**
  - some may already be in hand
  - some may not be easily fixable

# TOPCAT TAP Client

## TOPCAT v4.3 (08/2015): major TAP/ADQL client overhaul (some items enabled by post-TAP1.0 standard developments)

- Service discovery improvements — see Reg WG
- Metadata retrieval: pluggable & scalable
- Service metadata display: VOResource shown, UDFs shown
- Metadata display: scalable
- Examples: service-provided, ObsTAP, RegTAP
- Query editing: multi-tab, undo/redo, UDF-aware syntax checking
- Configurability options: metadata acquisition, upload/response VOTable serialization variant, service discovery, HTTP-level compression (mainly of interest to TAP geeks)
- ADQL cheat sheet (v4.3-1)
- Diagnostics: log non-VOTable error responses, log TAP query curl(1) equivalents

The screenshot shows the TOPCAT TAP Client interface. The main window is titled "Table Access Protocol (TAP) Query". It features a menu bar with "Window", "TAP", "Edit", "Interop", and "Help". Below the menu bar are several icons for navigation and actions. The interface is divided into several sections:

- Metadata:** A tree view on the left shows a search for "califa hipparcos". The tree includes "GAVO DC TAP (21/141)", "arihip (1/1)", "califadr1 (7/7)", and "califadr2 (7/7)". The "califadr1" folder is expanded, showing sub-folders like "cubes", "fluxpos", "fluxv12", "fluxv50", "objects", and "spectra".
- Table View:** A table on the right displays metadata for various columns. The columns are: Name, DataType, Indexed, Unit, and Description. The table lists columns such as "hipno", "srcsel", "raj2000", "dej2000", "pmra", "pmde", "t\_ra", "err\_ra", "err\_pmra", "t\_de", "err\_de", "err\_pmde", "parallax", and "e\_parallax".
- Service Capabilities:** A section below the metadata table shows "Query Language: ADQL-2.0", "Max Rows: 2000 (default)", and "Uploads: 20Mb".
- ADQL Text:** A text area for writing queries. The mode is set to "Synchronous". The query text is:

```
SELECT TOP 10000
  o.target_name, o.raj2000, o.dej2000, o.mag, o.magz,
  h.hipno, h.raj2000, h.dej2000, h.pmra, h.pmde
FROM califadr1.objects AS o
JOIN arihip.main AS h
ON 1=CONTAINS(POINT('ICRS', o.raj3000, o.dej2000),
  CIRCLE('ICRS', h.raj2000, h.dej2000, 5./3600.))
```
- Run Query:** A button at the bottom right to execute the query.

# TAP Status

- Since TAP v1.0 REC (2010):
  - More/better clients
  - Better services (server library developments, validator availability, experience with protocol)
  - More services registered
  - Enhancements to standards (RegTAP, TAPRegExt, /examples endpoint, more on the way)
- Usage
  - I am now approaching *“Not embarrassed”* to show TOPCAT/TAP to astronomers
    - ▷ you don't need to be a VO expert to use it any more
    - ▷ you do need to have a slight understanding of SQL
    - ▷ post-SDSS, many astronomers have at least a rough idea of SQL, but generally need/like a bit of help (*examples!*)
  - Astronomers can/should be using TAP to do science now
    - ▷ for single-archive queries
    - ▷ for multi-archive data integration(?)
      - are they?

# Service-Provided Examples

## Standard TAP service /examples endpoint

- Provides example ADQL queries specific to TAP service
- Examples document is XHTML marked up using RDFa in a standard way ... except there are two competing versions of the standard, DALI 1.1 vs. TAP Note
- This is a great way to help astronomers make good use of TAP

## Actions:

- DAL: Choose between DALI 1.1 and TAPNote standard example format. Soon!
- Service providers: provide examples
- IVOA/all: encourage service providers to provide examples (it's a recent addition, most don't know it exists)
- MBT: provide validator tool for examples (taplint enhancement)

# Examples Format

## XHTML/RDFa markup for service-provided examples:

- Options under consideration:

- ▷ DALI 1.1 (*better for protocol-agnostic automatic query generation*):

```
<div typeof="example" id="basicQuery" resource="#basicQuery">
  <h2 property="name">Basic Query</h2>
  <p>This is a simple query on the PhotoObj table:</p>
  <div property="generic-parameter" typeof="keyval">
    <span class="invisible" property="key">QUERY</span>
    <pre property="value" >SELECT * FROM sdss.PhotoObj</pre>
  </div>
</div>
```

- ▷ TAP Note (*easier for service providers to write*):

```
<div typeof="example" id="basicQuery" resource="#basicQuery">
  <h2 property="name">Basic Query</h2>
  <p>This is a simple query on the PhotObj table:</p>
  <h2 property="name">Basic Query</h2>
  <pre property="query">SELECT * FROM sdss.PhotoObj</pre>
</div>
```

- Currently, TOPCAT recognises either variant
- I (finally get off the fence and) weakly favour TAPNote, to lower the bar for service providers
- Or maybe these should be two different endpoints for different purposes?
- ... but really I just want a decision

# Scalable Metadata Acquisition

For VizieR-scale services, you can't download all metadata at once

- You can do it using multiple TAP\_SCHEMA queries

- ▶ This works mostly OK

```
SELECT column_name, description, ...  
FROM TAP_SCHEMA.columns  
WHERE table_name = 'xxx'
```

- ▶ ... except column list is unordered

- ▶ That's bad when presenting 500-column table metadata to the user

- new `column_index` column in `TAP_SCHEMA.columns`?

- There are proposals for multi-stage VODataService XML queries to `/tables` endpoint

- ▶ VizieR

- ▶ VOSI 1.1

- TOPCAT can use either on request, but using the wrong one gets misleading results

- Action: [DAL](#): standardise one scheme

# Positional Crossmatch

“Standard” ADQL positional xmatch is of the form:

```
JOIN ON 1=CONTAINS( POINT('ICRS', t1.ra, t1.dec),  
                   CIRCLE('ICRS', t2.ra, t2.dec, radius_in_deg))
```

- This is a very common thing for users to want to do
  - ▷ It's ugly and hard to remember
  - ▷ The coord-sys arguments are pointless but necessary
  - ▷ Geometry functions are optional — not all services support them
  - ▷ Some services support it but with very poor performance (not indexed?), and offer other/better ways to do a crossmatch
  - ▷ Is it really standard? Where is it written down that this is how you do xmatch?
  - ▷ It's embarrassing to tell astronomers that this is what you have to write
- Could it be improved? e.g.

```
JOIN ON 1=XMATCH(t.ra1, t.dec1, t.ra2, t.dec2, radius_in_deg)
```

  - ▷ It would look less nasty
  - ▷ It might allow implementations to offer positional crossmatch without implementing geometry functions (i.e. PGSphere requirement)
  - ▷ Could be prototyped as semi-standard syntax before/without ADQL revision
  - ▷ ... but maybe there's some fundamental reason it can't/shouldn't be done 😞



# Service Discovery

Hard to locate TAP services by dataset name from the Registry

- but you can do it by cheating (GloTS) — see my talk in Registry

# The Usual

- Correctness and compliance
  - Many services still broken/unreliable/slow
- Capabilities
  - Optional capabilities not always implemented (obviously)
    - ▷ Upload
    - ▷ Geometry functions
- Registration
  - Public services slow to arrive in Registry

# Conclusions

- TAP is in much better shape than a few years ago
  - Better standards, more services, better server libraries, better services, better clients, better validation
- Where upload and xmatch both implemented, it's very powerful
  - that doesn't seem to apply to too many services
- I'm *almost* not embarrassed to show it to astronomers
- Still some things I'd like to see addressed  
(some may be controversial, some already in hand):
  - standardise `/examples` format
  - standardise scalable metadata acquisition protocol
  - encourage service example provision
  - new/alternative positional cross-match syntax
  - new column `column_index` column `TAP_SCHEMA.columns`
  - improve service registration
  - improve service capabilities
  - improve service correctness

... Easy

... Hard