TAP Service Discovery from TOPCAT

Mark Taylor (Bristol)

IVOA Interop
Sydney

31 October 2015

$Id: tap-discovery.tex,v 1.10 2015/10/29 08:15:03 mbt Exp $

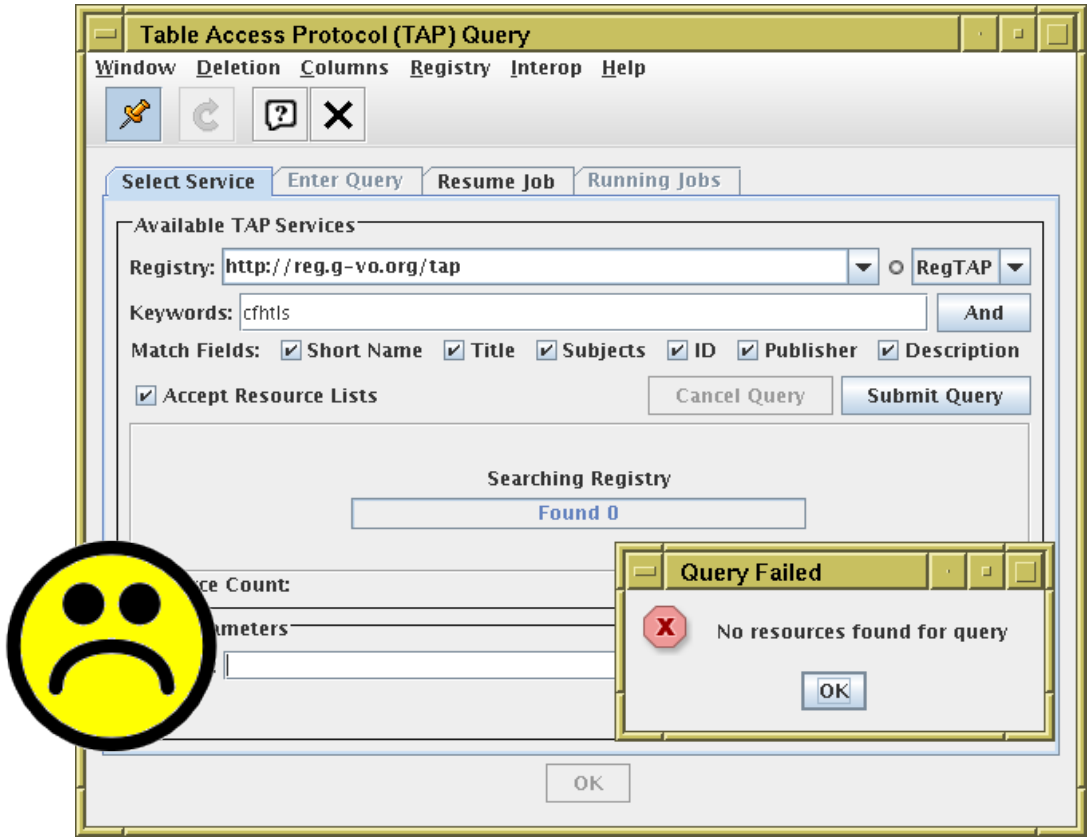
Outline

- TOPCAT requirements: locate TAP services by table metadata
- Implementations:
  - Vanilla registry
  - GloTS
  - Registry with auxiliary records
- Conclusions

Search by Service Metadata

Previous versions of TOPCAT:

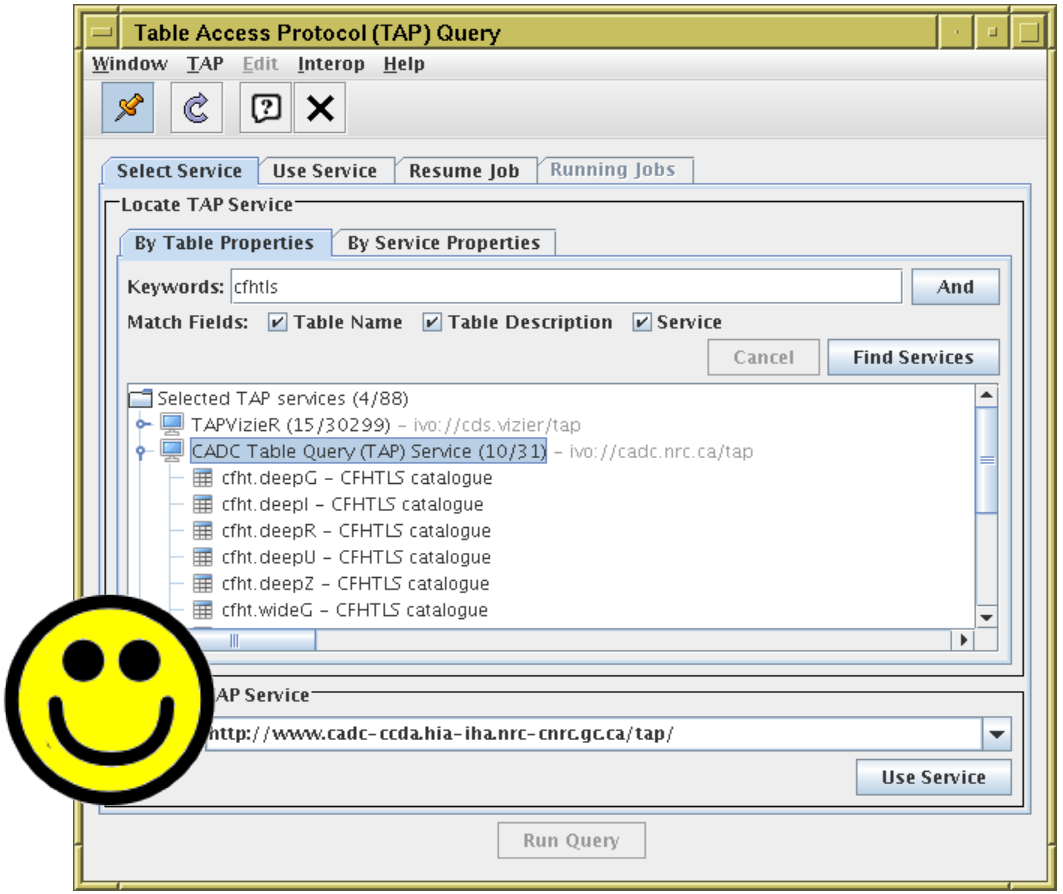
- Search terms matched against service VOResource metadata
  - ▷ Some combination of service metadata items:
    ShortName, Title, Subjects, IVOID, Publisher, Description
- Works if you know the service name
  (e.g. GAVO, HEASARC)
- Works if service happens to be named like dataset
  (some small, targetted TAP services
  e.g. SDSS, UKIDSS, CSC;
  mostly WFAU and JVO)
- No good for finding dataset by name in larger, multi-dataset TAP services
  (e.g. RAVE, Spitzer, UCAC4, CFHTLS, WMAP, FERMI, ... and most others)



Search by Table Metadata

Recent TOPCAT release (v4.3):

- Search terms matched against table VODataService metadata
  - ▷ Some combination of table Name, Description and service Name
- Works if you know the dataset name or description words
- Option to search by service name too
- Much better fit to how astronomers want to locate data



Discovery by Table Metadata: Requirements

TOPCAT GUI needs to:

- List all TAP services
  - ▷ get metadata (Name, Service URL) for each service
  - ▷ get table count for each service
- Find tables
  - ▷ restrict by constraint (table name and/or description matches given search terms)
  - ▷ link back to TAP service for each found table

Discovery by Table Metadata: Naive RegTAP

Search Registry for TAP services by table metadata:

```
SELECT ivoid, short_name, res_title, table_name
        FROM rr.res_table
NATURAL JOIN rr.resource
NATURAL JOIN rr.capability
NATURAL JOIN rr.interface
WHERE standard_id='ivo://ivoa.net/std/tap'
  AND (table_name LIKE '%cfhtls%' OR
       table_description LIKE '%cfhtls%')
```

Discovery by Table Metadata: Naive RegTAP

Search Registry for TAP services by table metadata:

```
SELECT ivoid, short_name, res_title, table_name
        FROM rr.res_table
NATURAL JOIN rr.resource
NATURAL JOIN rr.capability
NATURAL JOIN rr.interface
WHERE standard_id='ivo://ivoa.net/std/tap'
  AND (table_name LIKE '%cfhtls%' OR
       table_description LIKE '%cfhtls%')
```

Doesn't work well ☹

- Most TAP resource records lack "fine-grained" table metadata
  - ▷ (Probably just as well - fine-grained TAPVizieR registry record would be huge)

Discovery by Table Metadata: GloTS

GloTS to the rescue!

- Global TAP Schema is maintained by Markus at ARI (GAVO DC TAP)
- Auto-updated by regularly crawling TAP services and reading metadata (?)
- It contains info on all registered TAP services in 3 tables:
  - `glots.services` (98 rows)
  - `glots.tables` (34,426 rows)
  - `glots.columns` (168,442 rows)
- This is exactly what I need! (and more straightforward to query than RR)

```
SELECT ivoid, table_name, table_desc
FROM glots.tables
WHERE 1=ivo_nocasematch(table_name, '%cfhtls%') OR
      1=ivo_hasWord(table_desc, 'cfhtls')
```

Discovery by Table Metadata: GloTS

GloTS to the rescue!

- Global TAP Schema is maintained by Markus at ARI (GAVO DC TAP)
- Auto-updated by regularly crawling TAP services and reading metadata (?)
- It contains info on all registered TAP services in 3 tables:
  - `glots.services` (98 rows)
  - `glots.tables` (34,426 rows)
  - `glots.columns` (168,442 rows)
- This is exactly what I need! (and more straightforward to query than RR)

```
SELECT ivoid, table_name, table_desc
FROM glots.tables
WHERE 1=ivo_nocasematch(table_name, '%cfhtls%') OR
      1=ivo_hasWord(table_desc, 'cfhtls')
```

Works perfectly ☺

- GloTS is not a standard, but reliability is good
  - Now mirrored beween ARI and AIP (one day Paris?)
  - Manual DNS failover from reg.g-vo.org
  - Thanks Markus!!

Discovery by Table Metadata: Registry #aux

Search registry using additional Auxiliary TAP records

- Relies on joins with `rr.relationship` table, along with special `ivo://ivoa.net/std/tap#aux` standardIds:

```
SELECT DISTINCT related_id, table_name, table_description
FROM rr.relationship
NATURAL JOIN rr.capability
NATURAL JOIN rr.interface
NATURAL JOIN rr.res_table
WHERE (relationship_type = 'served-by' AND
       standard_id = 'ivo://ivoa.net/std/tap#aux' AND
       intf_type = 'vs:paramhttp')
  AND (1=ivo_nocasematch(table_name, '%cfhtls%') OR
       1=ivo_hasWord(table_description, 'cfhtls'))
```

Discovery by Table Metadata: Registry #aux

Search registry using additional Auxiliary TAP records

- Relies on joins with `rr.relationship` table, along with special `ivo://ivoa.net/std/tap#aux` standardIds:

```
SELECT DISTINCT related_id, table_name, table_description
FROM rr.relationship
NATURAL JOIN rr.capability
NATURAL JOIN rr.interface
NATURAL JOIN rr.res_table
WHERE (relationship_type = 'served-by' AND
       standard_id = 'ivo://ivoa.net/std/tap#aux' AND
       intf_type = 'vs:paramhttp')
  AND (1=ivo_nocasematch(table_name, '%cfhtls%') OR
       1=ivo_hasWord(table_description, 'cfhtls'))
```

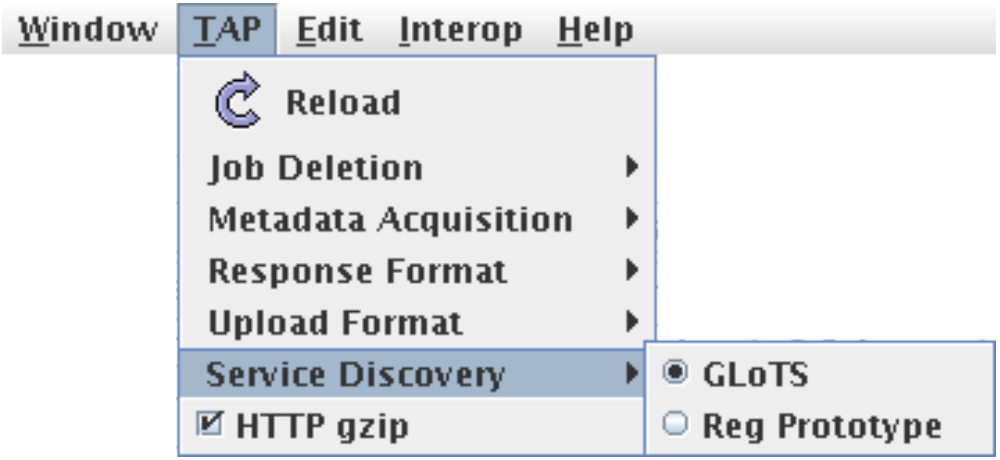
Works, but only on prototype services

- A couple of services registered like this in GAVO RR - works for those
- Otherwise, few (no?) `#aux` standardIds currently registered
- Table `rr.relationship` not always well-maintained

Implementation Status

Pluggable implementation in TOPCAT TAP client

- TAP by-table service discovery done via TapServiceFinder interface
- Currently implementations for GloTS and Aux Prototype
- Other implementations could be added fairly easily for experimentation
- GUI switch hidden away in **TAP|Service Discovery** menu; only intended for TAP hackers
- Uses GloTS by default
- If Registry grows reliable By-Table discovery in the future, that could be the default



Conclusions

- GloTS works really well
  - ... as long as Markus maintains it
  - ... but it feels a bit like cheating (bypasses registry, not standard)
- Registry #aux standard IDs could in principle work
  - Proof of concept working
  - ... but relies on service providers to register services in not-totally-obvious way
  - ... which is probably a long way off
- TOPCAT implementation:
  - provides user choice between available service discovery options
  - but will default to whatever works best at release time