TAP Specification

- current draft is TAP 0.3, recent work by editors only
- agreements from Trieste, 2008
- scope:
  - sync and async query execution
  - ADQL queries and param-based queries
- balance UWS and DAL
  - UWS: REST, resource-based, async
  - DAL: operation-based, sync
  - compliance requirements: MUST and SHOULD

TAP Specification

- service base URL ($baseURL): not specified
- http endpoints
  - $baseURL/async (MUST)
  - $baseURL/sync (MUST)
- core DAL params: REQUEST, VERSION
  - REQUEST=adqlquery (MUST)
  - REQUEST=paramquery (SHOULD)
  - VERSION=<version of the TAP spec>
  - REQUEST and VERSION tell service how to interpret remaining params

TAP Specification

- async ADQL query:
  - POST http://example.com/path/async?REQUEST=adqlquery&...
  - response: redirect to job URL

  ...
- sync ADQL query:
  - POST, GET? http://example.com/path/sync?REQUEST=adqlquery&... (all params specified here)
  - response: blocks, returns result

TAP Specification

- async PARAM query:
  - POST http://example.com/path/async?REQUEST=paramquery&...
  - response: redirect to job URL

  ...
- sync PARAM query:
  - POST, GET? http://example.com/path/sync?REQUEST=paramquery&... (all params specified here)
  - response: blocks, returns result

TAP Specification: async

- UWS pattern
  - POST creates a new query in PENDING state, redirects to http://example.com/path/async/<jobID>
  - GET returns the job-list
  - all resources under the job are mutable (PENDING)
  - execute job by POST PHASE=RUN to http://example.com/path/async/<jobID>/phase
  - monitor phase resource until job done (COMPLETED, ERROR, ABORTED)

async job resources

- following UWS, each parameter is exposed as a resource of the same name, e.g.
  - $baseURL/async/<jobID>/request
  - $baseURL/async/<jobID>/version
  - $baseURL/async/<jobID>/query
  - $baseURL/async/<jobID>/lang
  - $baseURL/async/<jobID>/format
  - $baseURL/async/<jobID>/upload/foo
  - $baseURL/async/<jobID>/upload/bar
  - plus the standard UWS resources
- need a simple schema for these

REQUEST=adqlquery

- QUERY=<ADQL query> (MUST)
  - if tables contain datetime values, support ISO8601 datetime format (MUST)
  - if tables contain spatial values, support POINT, CIRCLE, BOX, REGION, INTERSECTS, COORDSYS, COORD1, COORD2 (MUST)
  - support STC-S as string format for REGION (MUST)
  - support STC-S coordinate systems in POINT, CIRCLE, BOX (MUST)
  - support AREA, CENTROID, CONTAINS, POLYGON (MAY)
- LANG=<query language and version> (MUST)

REQUEST=adqlquery (and param)

- FORMAT=votable (MUST)
- FORMAT=csv (SHOULD)
- UPLOAD=<table name>,<table URI>;...
  - table name must be a legal ADQL table name
  - table URI (vos:, http:, etc) provides the content
  - tables usable in query using specified name, in reserved schema TAP_UPLOAD, e.g.
    - UPLOAD=foo,http://example.com/foo.xml
    - SELECT * from TAP_UPLOAD.foo
  - column names from VOTable FIELD element, name attribute: must be legal ADQL column names

multi-position query

POST $baseURL/async?

REQUEST=adqlquery&

UPLOAD=bar,http://example.com/mytable.xml&

QUERY=SELECT * FROM atable AS a JOIN TAP_UPLOAD.bar AS b
ON INTERSECTS(a.shape, b.location)

...

- all the same params with async and sync endpoints
  - endpoints could have different limits (MAXREC)

outstanding issue: metadata

- capabilities, availability, table metadata
- DAL2: REQUEST=getCapabilities
  - which endpoint? both?
  - table metadata identical for both endpoints
  - capabilities identical?
  - availability always the same?
- REST: $baseURL/capabilities
- SSA did not anticipate multiple endpoints
- sync + async (UWS) requires it