Leibniz-Institut für
Astrophysik Potsdam

uws-client

A command-line tool for UWS services

Kristin Riebe

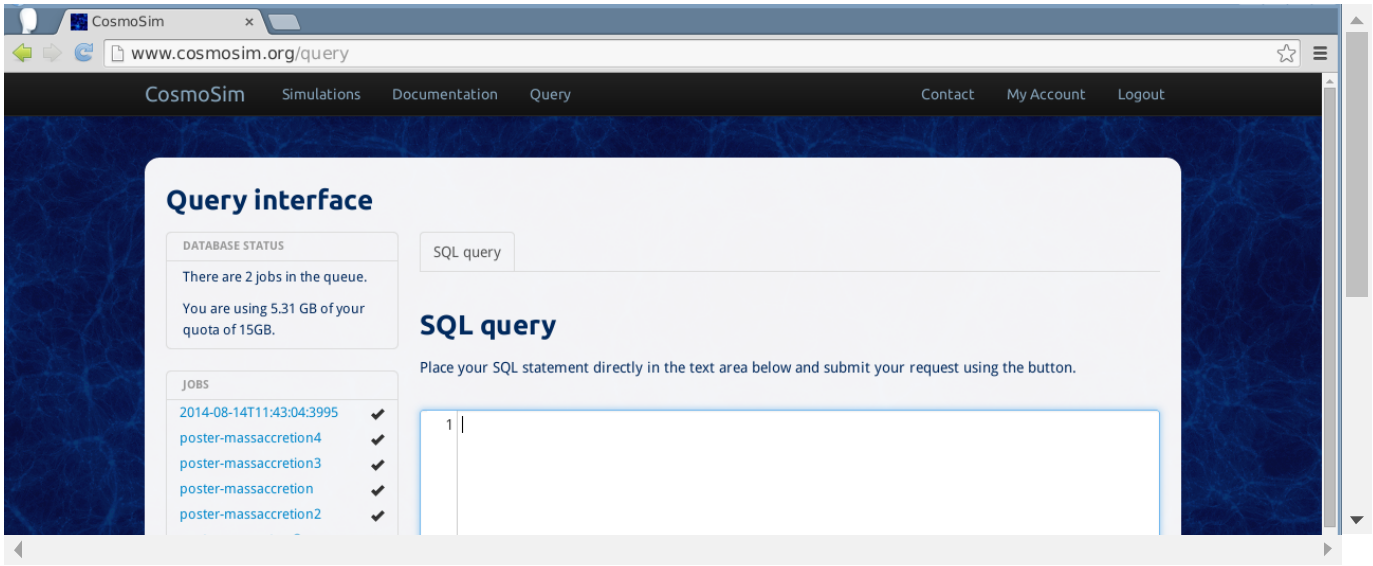
GAVO, AIP

UWS – Universal Worker Service

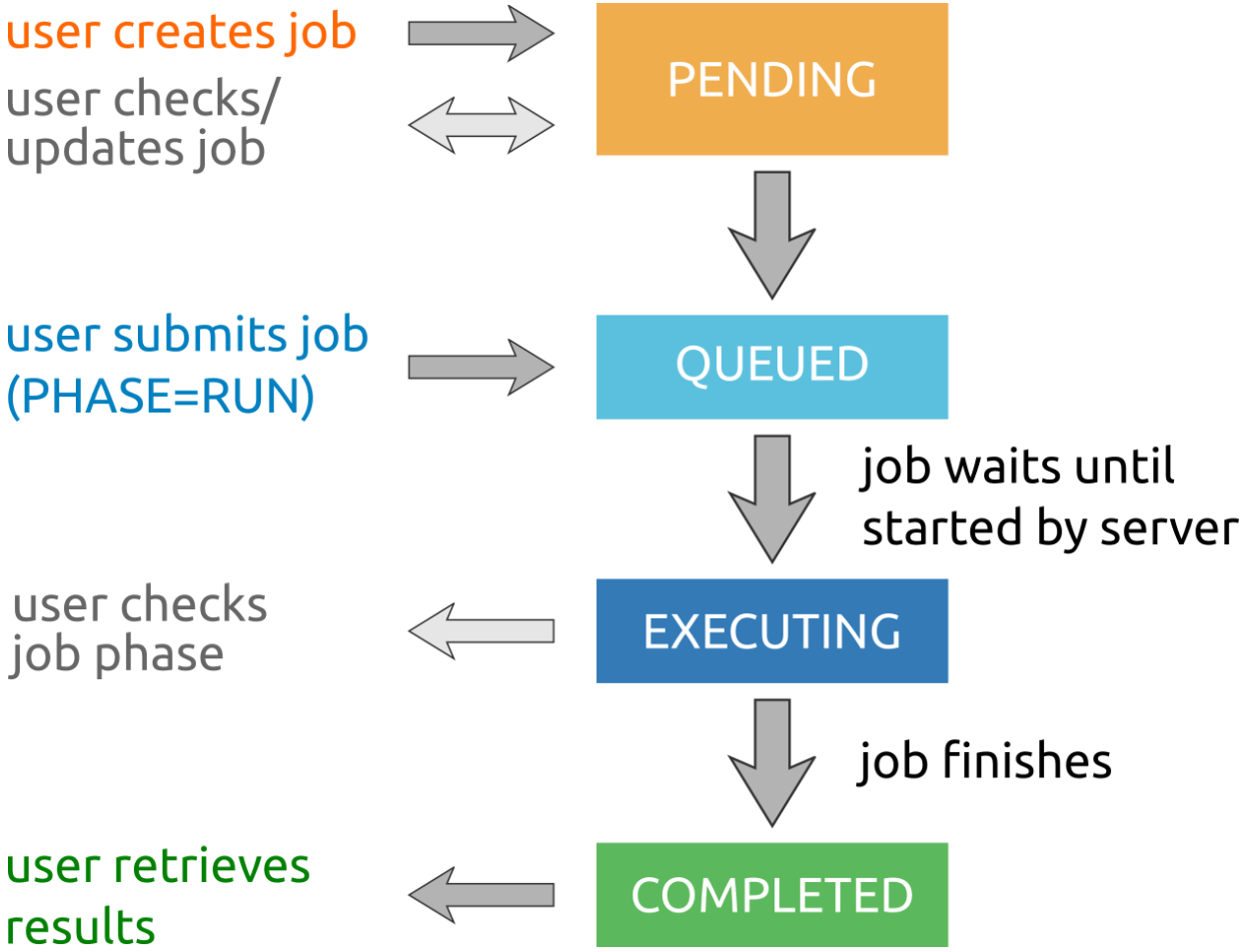
- VO standard: http://www.ivoa.net/documents/UWS/;
- latest development version on Volute-repository: http://volute.g-vo.org/svn/trunk/projects/grid/uws/doc/
- pattern for asynchronous, job-oriented web services
- used for asynchronous part of Table Access Protocol (TAP)

Example UWS service: CosmoSim

- cosmological simulation database
- web service with queueing system for SQL queries
- queries sent as jobs to database server
- results stored in user tables on database



Typical "job flow"



Why use UWS via command line?

- users want scripted access to services
  - e.g. for sending thousands of jobs
- integrate requests to the UWS service in own tools/pipeline/...
- What about TAP clients like Topcat, STILTS or tapsh?
  - perfect for TAP services
  - but not each UWS service is part of TAP

Simple http client

- UWS rest interface: GET, POST and DELETE requests
- httpie: http://httpie.org/
  - simple client for http requests
  - `pip install httpie`
  - Get job list:

    `http [...] https://www.cosmosim.org/uws/query`
  - Create job: POST request with job parameters

    `http [...] --form --follow POST`

    `https://www.cosmosim.org/uws/query`

    `query="SELECT x,y,z FROM MDR1.FOF LIMIT 10"`

Replace `[...]` by e.g. `--auth cosmodemo:gavo`

Job list: xml response [extract]

```
<?xml version="1.0" encoding="UTF-8"?>
<uws:jobs xmlns:uws="http://www.ivoa.net/xml/UWS/v1.0" xmlns:xlink="http:/
  <uws:jobref id="1445885741809796726" runId="" ownerId="uwstest" creation
    <uws:phase>PENDING</uws:phase>
  </uws:jobref>
  <uws:jobref id="1445889777454191483" runId="" ownerId="uwstest" creation
    <uws:phase>ABORTED</uws:phase>
  </uws:jobref>
  <uws:jobref id="1445892917374741131" runId="" ownerId="uwstest" creation
    <uws:phase>ERROR</uws:phase>
  </uws:jobref>
  <uws:jobref id="1445892943877819213" runId="" ownerId="uwstest" creation
    <uws:phase>ABORTED</uws:phase>
  </uws:jobref>
```

More comfortable: uws-client

- uws-client: https://github.com/aipescience/uws-client
  - `git clone https://github.com/aipescience/uws-client`
  - `cd uws-client`
  - `pip install .`
- written by **Adrian Partl**
- UWS 1.1 updates by me
- library with python classes for job and job list
- wrapper for http-requests
- pretty formatting of output

Example: Get job list

- with uws-client:
  - `uws --user cosmodemo --password gavo --host https://www.cosmosim.org/uws/query list`
  - filter jobs by phase, e.g. all jobs with ERROR:

    `uws [...] list --error`

Job list formatted by uws-client

```
List of jobs on UWS service for user: 'uwstest'
      Job Id                    Status
================================================
1445885741809796726             PENDING
1445889777454191483             ABORTED
1445892917374741131             ERROR
1445892943877819213             ABORTED
4 jobs listed.
```

For new services, UWS1.1:

```
List of jobs on UWS service for user: 'uwstest'
      Job Id                    [Run]             [Owner]        [Creation T
================================================================================
1445885741809796726                                 uwstest    2016-05-02T14:05
1445889777454191483                                 uwstest    2016-05-02T14:05
1445892917374741131                                 uwstest    2016-05-02T14:05
1445892943877819213                                 uwstest    2016-05-02T14:05
4 jobs listed.
```

Basic commands

- Create a job
  - `uws [...] job new query='SELECT ....'`
- Start the job
  - `uws [...] job run <jobid>`
- View job details, check phase
  - `uws [...] job show <jobid>`
- Retrieve the results
  - `uws [...] job results <jobid> <resultid>`

Replace `[...]` by e.g. `--user cosmodemo --password gavo --host https://www.cosmosim.org/uws/query`

Job details from uws-client

```
Field                    Value
=========================================================
Job id                   369123305288395
Owner id                 cosmodemo
Phase                    COMPLETED
Start time               2015-09-10T14:25:47+02:00
End time                 2015-09-10T14:25:47+02:00
Execution duration       30
Destruction time         2999-12-31T00:00:00+01:00
Parameter database       cosmosim_user_cosmodemo
Parameter table          2015-09-10-14-25-42-2328
Parameter query          SELECT x,y,z FROM MDR1.FOF LIMIT 10 [.
Parameter queue          short
Result csv               https://www.cosmosim.org/query/downloa
Result votable.xml       https://www.cosmosim.org/query/downloa
```

Job list filter

- filter by phase:
  - `uws [...] list --executing`
  - just append name of phase(s)
  - for UWS 1.0 (client-side filtering) and 1.1 (server side filtering)
- filter by time:
  - `uws [...] list --after 2016-01-01`
  - `uws [...] list --last 10`

WAIT

- Wait blocking behaviour:
  - when submitting job, add `--wait 100` for waiting 100 seconds before returning or until the phase is changing
  - `--wait -1` : wait forever (unless in final state)
  - `--wait 0` : do not wait
  - `--wait 100 --phase executing` : wait 100 seconds until phase change, but only if job is currently in executing phase
  - only if phase is active (PENDING, QUEUED, EXECUTING)

UWS discovery?

- Finding base-url for UWS is not automated, need to read documentation/ask service providers
- Will improve in the future?
  - improved registry records?
  - PDL for describing parameters?

Example services with UWS

- CosmoSim, RAVE, Applause plate archive (Potsdam)

  https://www.cosmosim.org/uws/query

  https://www.rave-survey.org/uws/query

  https://www.cosmosim.org/gaia.aip.de/uws/query

- UWS service for CTA

  https://voparis-uws-test.obspm.fr/

Example UWS endpoints for full TAP services

- Millennium (Garching)

  http://galformod.mpa-garching.mpg.de/millenniumtap/async

- CADC (Canada)

  http://www.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/tap/auth-async

- DaCHS (Heidelberg)

  http://dc.zah.uni-heidelberg.de/tap/async