UWS Server at VOPDC

Mathieu Servillat, Cyril Chauvin

Observatoire de Paris
VO-Paris Data Center

Computation at Observatoire de Paris

- **Tycho work cluster**
  - **16 nodes** : tycho[01-16]
    - 16 cores, Intel Xeon 2.60 GHz / 64 Go mem/node / 1,7 To disk space
  - **12 nodes** : quadri[17-28]
    - 8 cores, Intel Xeon 2.27 GHz / 24 Go mem/node / 160 Go disk space

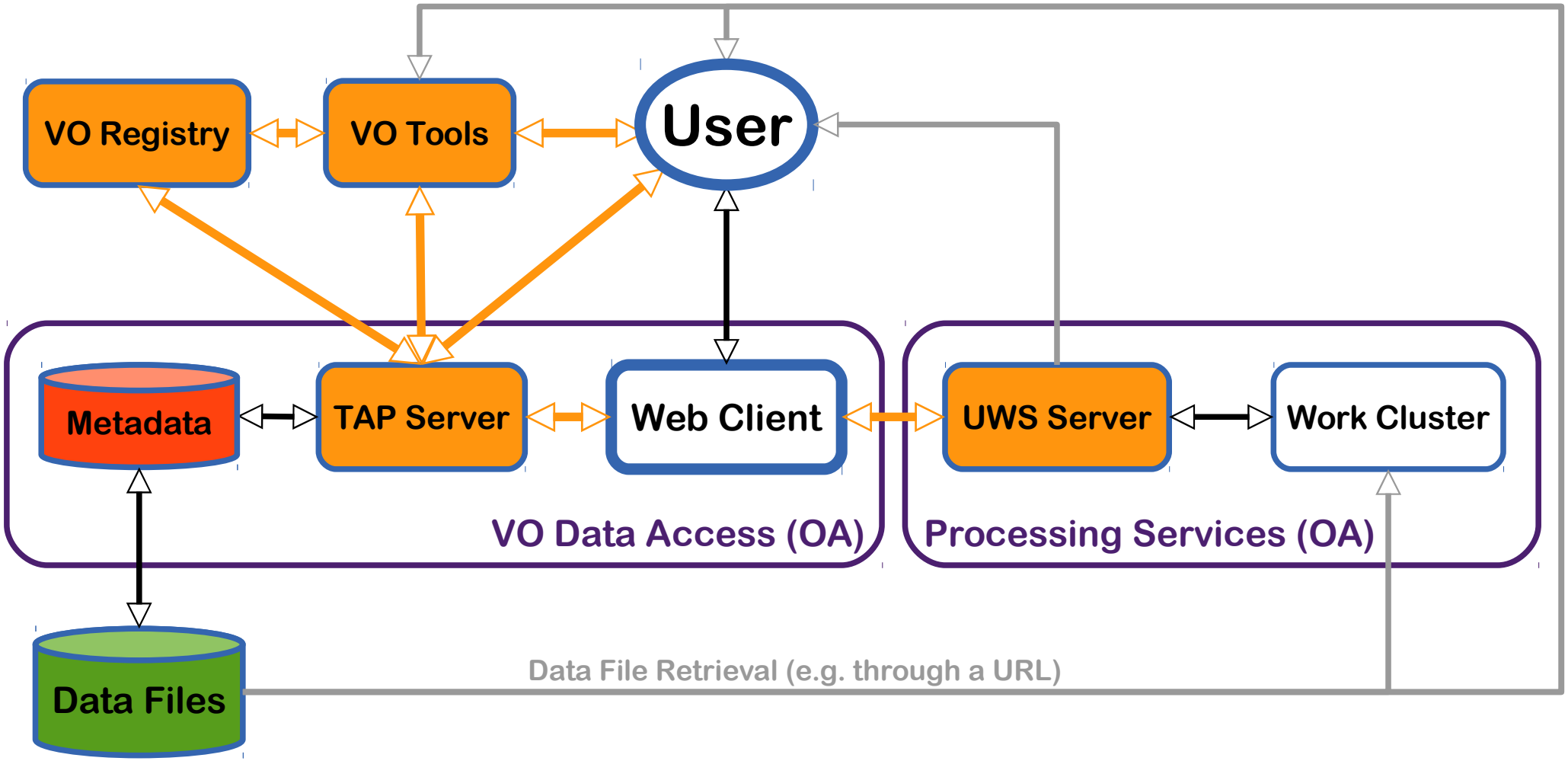
- **Simple Linux Utility for Resource Management**
  - **Manage resources**
    - Job execution
    - Job limitations /node/user
    - Node extinction
  - **Job Scheduler**
    - Backfill, fairshare, priority, preemption

Job Management at VOPDC

- **Specific context**
  - Work cluster (Tycho)
  - Job scheduler (SLURM)

- **Needs for VOPDC projects**
  - **Web based clients**
    - Data processing jobs
    - Wrap simulation codes
  - **Interface** to computational resources
    - Using VO Universal Worker System

Data Access for CTA (Cherenkov Telescope Array)

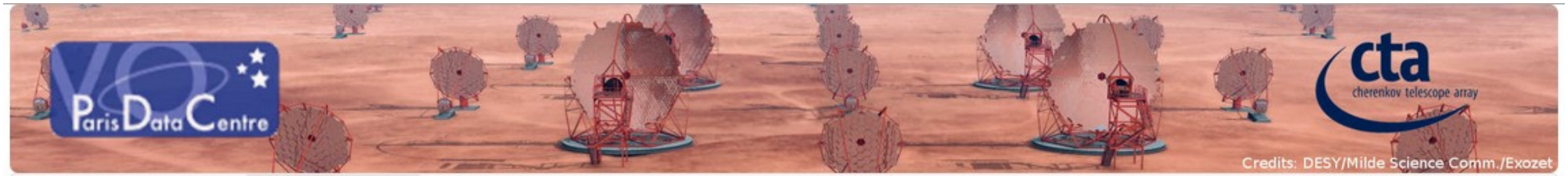


VO compliant Service
VO protocol
Database

CTA Data Model
CTA Archive
OA CTA Observer Access

Web Client

http://voparis-cta-client.obspm.fr



CTA Data Distiller | Search Form | Job List | Sign out **user**

Cone Search

Target Name	Crab Nebula
Source RA (deg)	83.633
Source Dec (deg)	22.514
Search radius (deg)	0.001

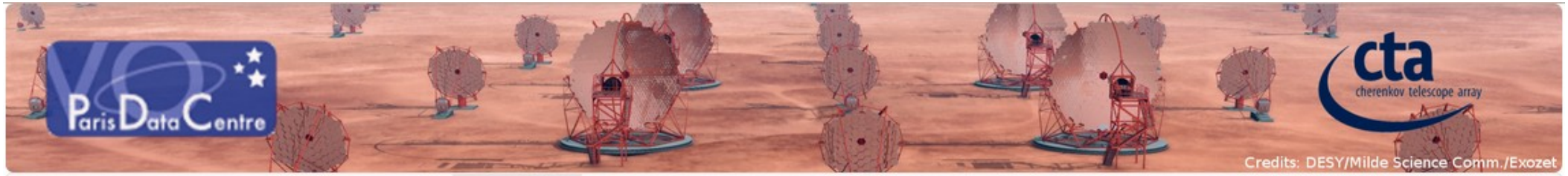
Submit Reset

Used to query Simbad with Sesame and set RA/Dec.

- Django, jQuery, BootStrap3
- Name resolver
  Simbad through Sesame
- Builds and Sends the ADQL query

ObsCore Search

proposal_id		Proposal ID
dataproduct_type	Nothing selected	Data product (file content) primary type
dataproduct_level	Nothing selected	DL0-5



CTA Data Distiller | Search Datasets | Results | Job List | Selected Job | JS9 | Authentication: Sign out **user**

Search | Analyse | Visualisation

SAMP

Interop (SAMP)
- Send Result Table
- Send Selected Data

Results

show/hide query

```
SELECT * FROM cta.vo_obscore as o WHERE 1 = intersects(o.s_region, circle('ICRS', 83.63308333, 22.0145, 0.001))
```

ADQL query

Send

ObsCore fields

Search

	dataproduct_type	obs_collection	obs_id	target_name	s_ra (deg)	s_dec (deg)
☐	eventlist	1	23592	Crab Nebula	82.01333618164062	22.01444435119629
☐	eventlist	1	23559	Crab Nebula	85.25333404541016	22.01444435119629
☐	eventlist	1	23526	Crab Nebula	83.63333129882812	22.51444435119629
☐	eventlist	1	23523	Crab Nebula	83.63333129882812	21.51444435119629
☐	eventlist	3	5003499	CrabNebula	83.28087615966797	21.784133911132812

UWS

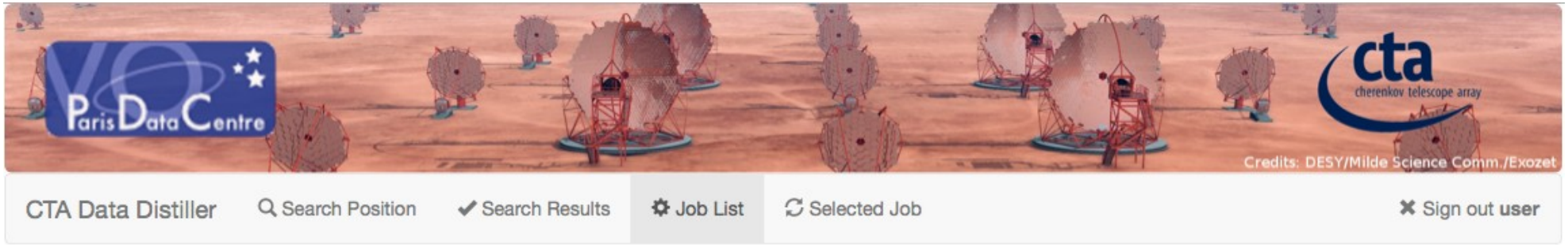
Showing 1 to 5 of 10 rows 5 records per page

<< < 1 2 > >>

Analysis tools
- Create Count Map(s)
- Extract Spectrum

Plotting tools
- TOPCAT
- Aladin
- VOSpec
- SPLAT

Web Client – Job List



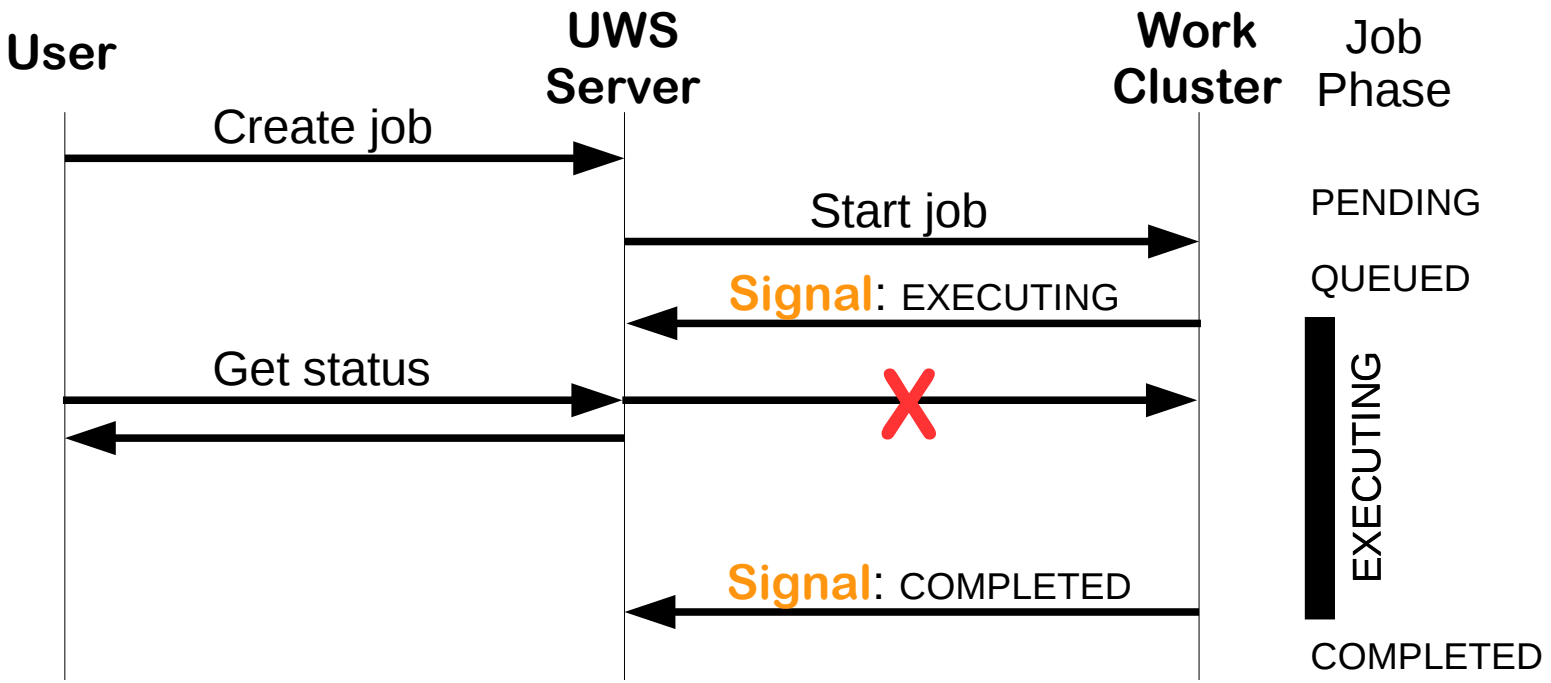
- UWS client using VOPDC **JavaScript library**
- UWS server...

New UWS server implementation

- **REST binding**
  - Using **bottle.py** Python micro-framework
- UWS **Job** attributes defined in a class
  - Methods to create, update, export (as XML)
- **Storage** classes
  - SQLiteStorage class implemented
  - Children classes can be added (PostgreSQL, NoSQL, File)
- **Manager** classes
  - SLURMManager class implemented
  - Children classes can be added (LocalManager, MoabManager...)

UWS server features

- **Separate job description from work cluster**
  - Wait for work cluster signals/events
  - Avoid (too many) status queries to work cluster



UWS server features

- **Full description of the UWS web service**
  - One **WADL** file (Web Application Description Language)
  - Describe parameters and results
  - Auto-generate parameter forms, results access
  - Test if submitted parameters are valid

```
−<application xsi:schemaLocation="http://wadl.dev.java.net/2009/02 http://www.w3.org/Submission/wadl/wadl.xsd">
  <doc>Implements the UWS 1.0 service</doc>
 −<grammars>
    <include href="http://ivoa.net/xml/UWS/UWS-v1.0.xsd"/>
  </grammars>
 −<representation id="parameters" mediaType="application/x-www-form-urlencoded">
    <!-- Job parameters for ctbin -->
   −<param style="query" name="evfile" type="xs:string" required="true" default="events.fits">
      <doc>Input event list or observation definition file</doc>
    </param>
   −<param style="query" name="outfile" type="xs:string" required="false" default="cntmap.fits">
      <doc>Output counts map or observation definition file</doc>
    </param>
   −<param style="query" name="prefix" type="xs:string" required="false" default="cntmap_" choices="0">
```

UWS server features

- **Python powered**
  - Logging of all activities with `logging`
  - Unit testing with `unittest` and `webtest`
  - Light, reusable, extensible code

- **Open source development**
  - voparis-git.obspm.fr with gitolite

UWS Standard comments

- **Not used in the v1.0 implementation**
  - PENDING barely used
    (Client sends all parameters and starts job)
  - HELD not necessary (managed by SLURM)
  - SUSPENDED not yet included (managed by SLURM)
- **Some redundancy**
  - start, delete, set parameters
  - But easy to implement
- **To be implemented (v1.1 and more)**
  - Pagination
  - Filters by phase
  - WAIT= (though not critical in our case)
  - Authentication system (using SSO/Shibboleth or HTTP auth)
  - Connection with Provenance DM and DataLink ?

IVOA Provenance DM (based on W3C)