

# Tracking provenance information with the OPUS job controller

Mathieu Servillat

Observatoire de Paris - LUTH  
Paris Astronomical Data Centre

IVOA Santiago - Oct. 2017



# OPUS: Observatoire de Paris UWS Server

## Main features

- IVOA standards
  - Universal Worker System (**UWS**)
  - **Provenance** data model
- REST architecture
  - Python micro-framework: `bottle.py`
- Collaborative development
  - Git server at PADC (`gitolite`)
  - GitHub: <https://github.com/mservillat/OPUS>
- Tests and quality
  - Unit tests with `unittest` and `webtest`
  - Activity history with `logging`

## Prototype available

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



# Job management at PADC

## Available structure

- Work cluster (Tycho)
- Job scheduler (SLURM)



## PADC projects

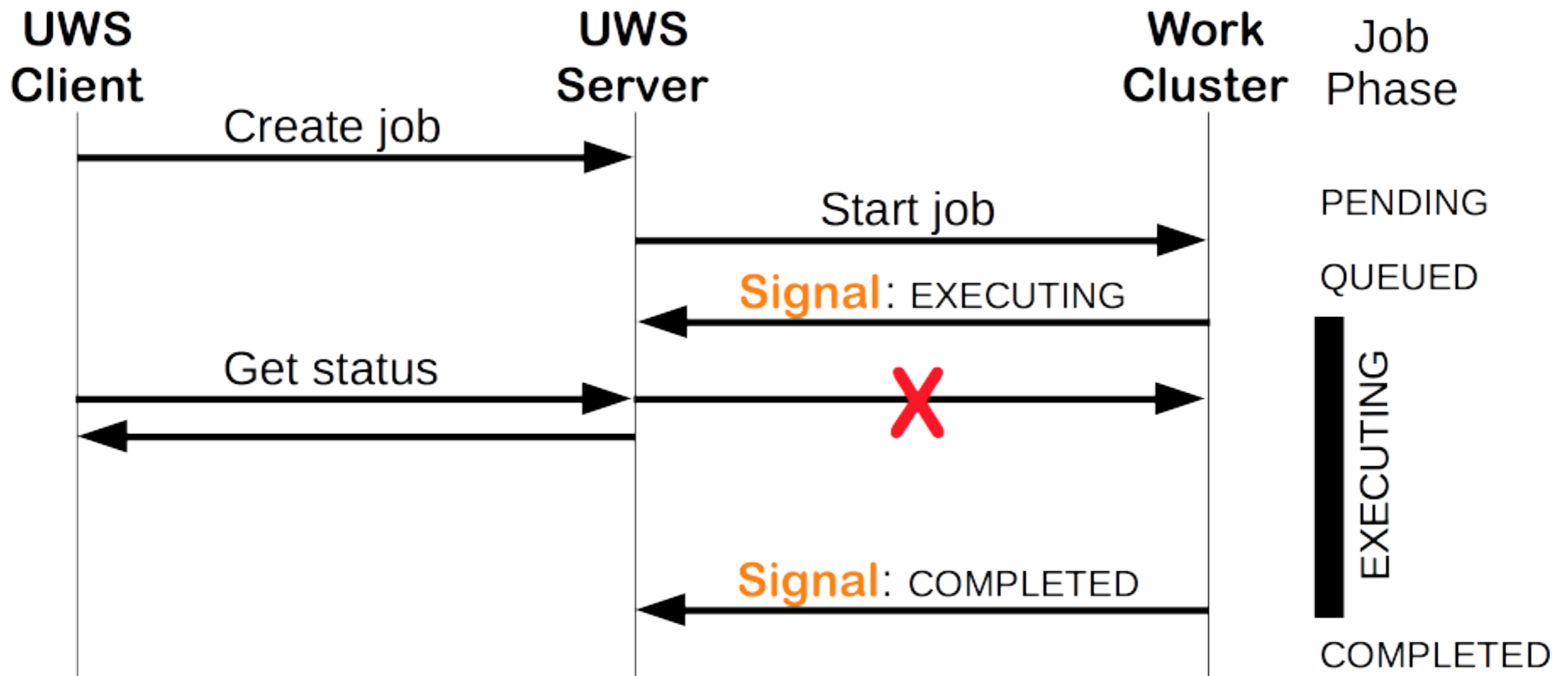
- Web based clients
  - Data access
  - Online data processing
  - Wrap simulation codes



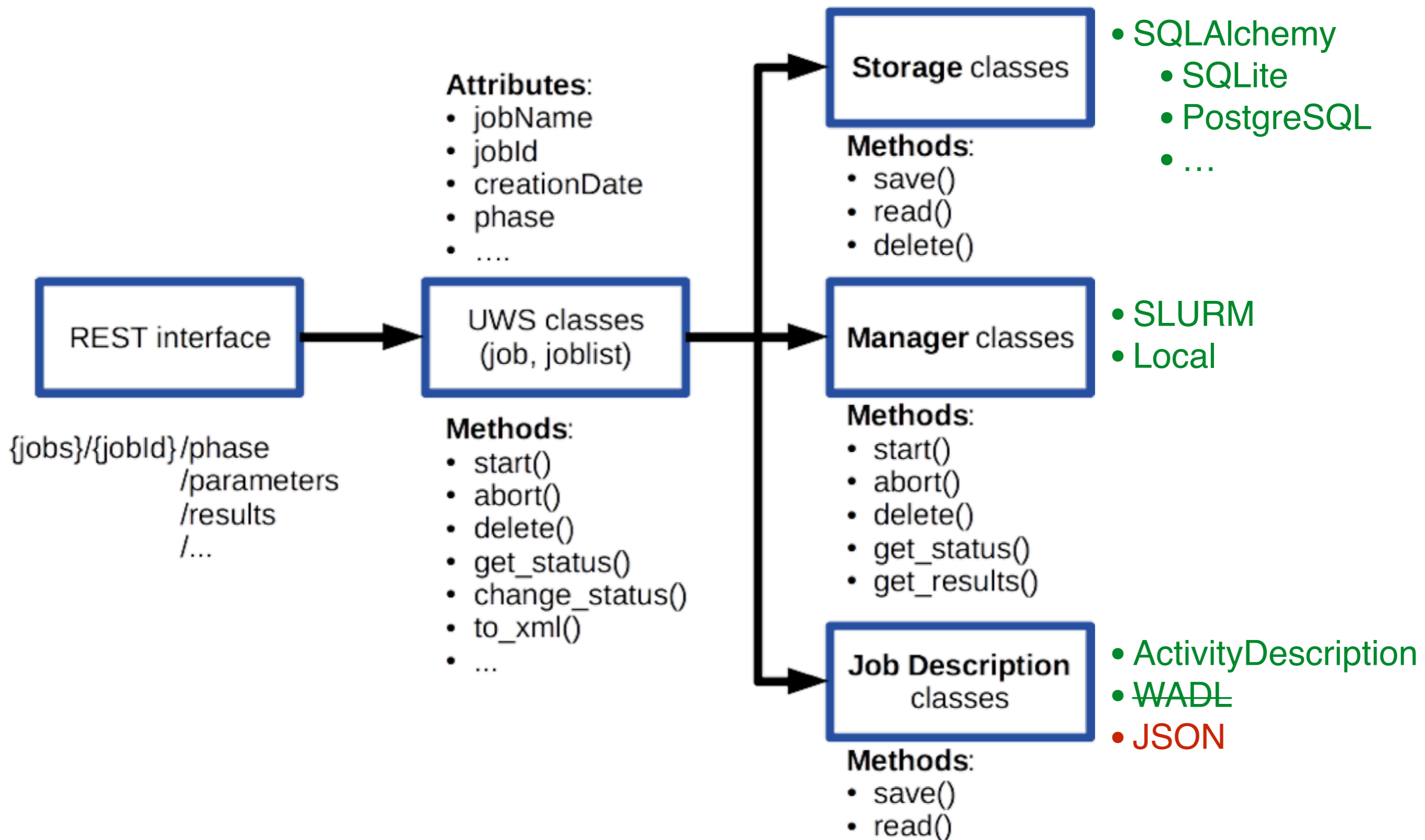
**Need a simple interface to computational resources**

# Inner workings

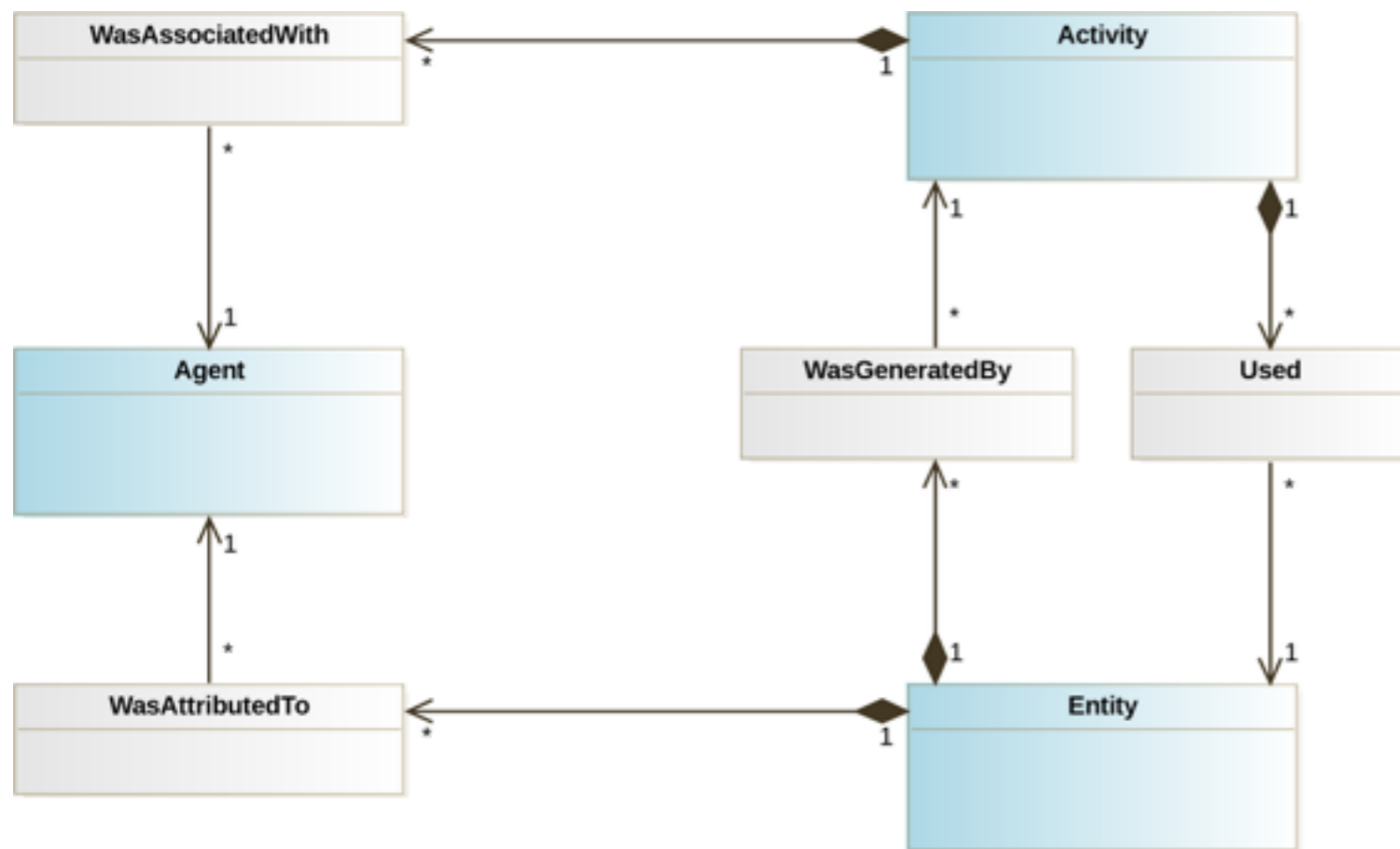
- Separate **job controller** from **work cluster**
  - Wait for work cluster **signals**
  - Avoid (too many) status queries to work cluster



# Server main classes



# Core Provenance Data Model



<http://www.w3.org/TR/prov-overview/>

30 April 2013



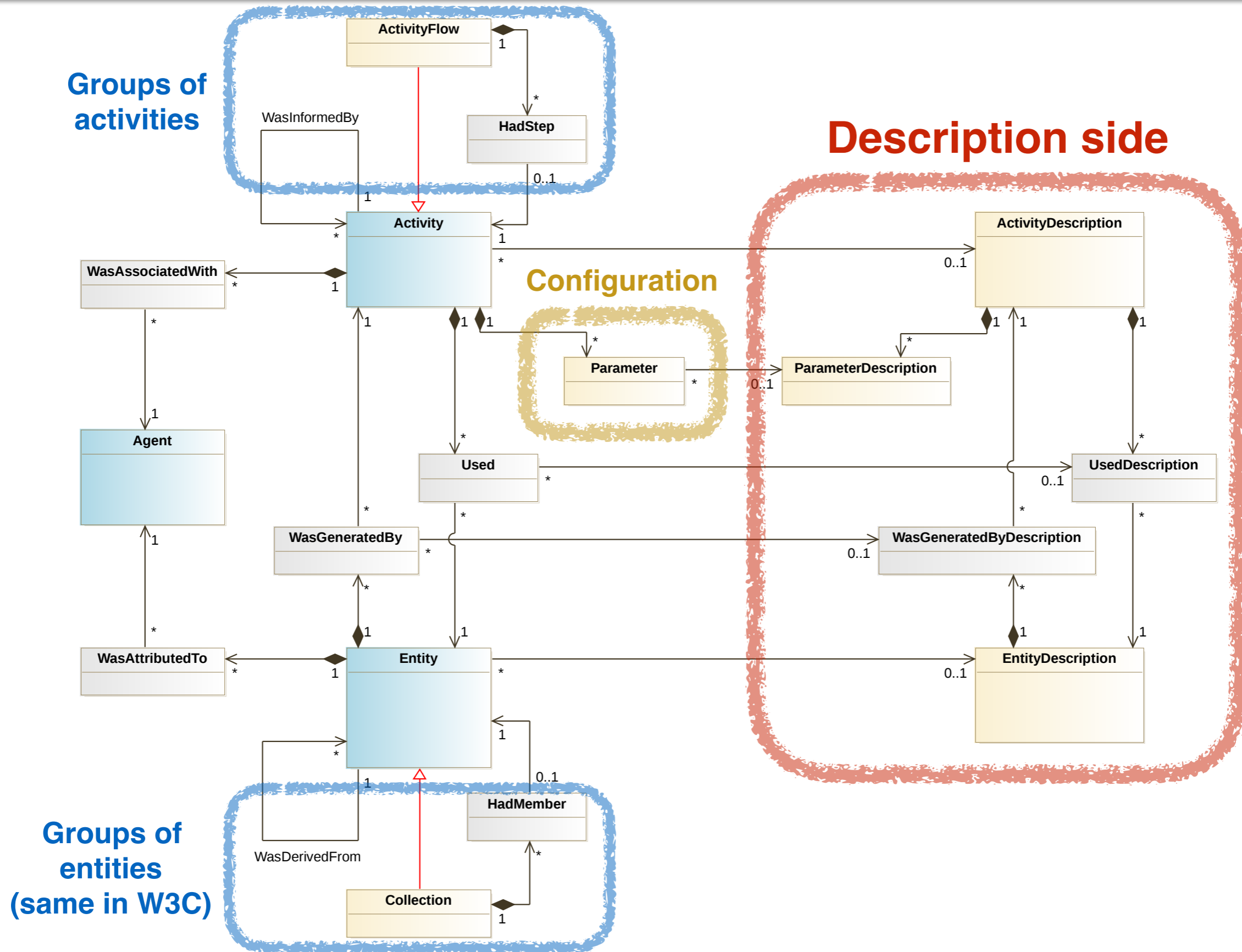
World Wide Web Consortium

<http://www.ivoa.net/documents/ProvenanceDM/>



- Core concepts from the W3C PROV recommendations
  - **Entity - Activity - Agent**
  - **Relations** and **roles** = provenance information
  - W3C PROV has many more relations
  - IVOA Provenance connected to **VO concepts** and **astronomy needs**

# IVOA Provenance Data Model diagram



# Serializations - ActivityDescription

```
<?xml version="1.0" encoding="UTF-8"?>
<VOTABLE xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://www.ivoa.net/xml/VOTable/v1.3" version="1.3"
  xsi:schemaLocation="http://www.ivoa.net/xml/VOTable/v1.3 http://www.ivoa.net/xml/VOTable/v1.3">
  <RESOURCE ID="make_RGB_image" name="make_RGB_image" type="meta" utype="voprov:ActivityDescription">
    <DESCRIPTION>Create an RGB image from 3 images</DESCRIPTION>
    <PARAM name="name" datatype="char" arraysize="*" value="make_RGB_image" utype="voprov:ActivityDescription.label" />
    <PARAM name="type" datatype="char" arraysize="*" value="..." utype="voprov:ActivityDescription.type" />
    <PARAM name="subtype" datatype="char" arraysize="*" value="..." utype="voprov:ActivityDescription.subtype" />
    <PARAM name="version" datatype="float" value="..." utype="voprov:ActivityDescription.version" />
    <PARAM name="doculink" datatype="char" arraysize="*" value="..." utype="voprov:ActivityDescription.doculink" />
    <PARAM name="contact_name" datatype="char" arraysize="*" value="..." utype="voprov:Agent.name" />
    <PARAM name="contact_email" datatype="char" arraysize="*" value="...@..." utype="voprov:Agent.email" />

    <GROUP name="InputParams" utype="voprov:ParameterDescription">
      <PARAM ID="RGB" arraysize="*" datatype="char" name="RGB" type="no_query" value="RGB.jpg">
        <DESCRIPTION>RGB image name</DESCRIPTION>
      </PARAM>
    </GROUP>

    <GROUP name="Used" utype="voprov:UsedDescription">
      <GROUP name="R" utype="voprov:EntityDescription">
        <DESCRIPTION>Image for red channel</DESCRIPTION>
        <PARAM name="default" value="R.jpg" arraysize="*" datatype="char" utype="voprov:Entity.id" />
        <PARAM name="role" value="red" arraysize="*" datatype="char" utype="voprov:UsedDescription.role" />
        <PARAM name="content_type" value="image/jpeg" arraysize="*" datatype="char" utype="voprov:EntityDescription.content_type" />
      </GROUP>
    </GROUP>

    <GROUP name="Generated" utype="voprov:WasGeneratedByDescription">
      <GROUP name="RGB" utype="voprov:EntityDescription">
        <DESCRIPTION>RGB image generated</DESCRIPTION>
        <PARAM name="role" value="RGB" arraysize="*" datatype="char" utype="voprov:WasGeneratedByDescription.role" />
        <PARAM name="content_type" value="image/jpeg" arraysize="*" datatype="char" utype="voprov:EntityDescription.content_type" />
      </GROUP>
    </GROUP>

  </RESOURCE>
</VOTABLE>
```

—> VOTable PARAM + attributes

—> DataLink Service Descriptor + Used/Generated groups



# Client main features

- Javascript based + Python Flask
  - `UwsLib.js`: sends AJAX requests to the server
  - `uws_manager.js`: handles and displays responses
    - Integration with Bootstrap3
    - HTML pages with specified `<div>` elements (id=joblist, parameters, results...)
- **Job definition editor**
  - Interface to create the ActivityDescription file
    - parameters, used and generated entities
    - bash execution script
- **Job manager**
  - list jobs, create jobs, control jobs, view results

# Client - Job definition

OPUS Job Definition Job List Signed in as admin

Job Definition Validate Job Copy script

**Name**  Load JDL Get JDL Job name.

**Description**  Job description.

**URL**  Job URL.

**Contact name**  Job contact name.

**Contact email**  Job contact email.

**Input**

obsids	=	47802 47803 47804 47827 47	image/fits	↑	↓	×
Desc. List of runs						
File <input type="radio"/> or ID <input checked="" type="radio"/> and URL <input type="text" value="http://url_to_the_input_file?id=\$ID"/>						
<span>Add input</span> <span>Remove all input</span>						

List of input entities (e.g. files) used with their name and content type. The input is a File or an ID, possibly with a URL to resolve the ID and download the file (use \$ID in the URL template). If no URL is specified, the script itself should be able to resolve the ID and get the file. Note that an input can refer to a parameter (if it has the same name), e.g. the name of an input file used in the script.

**Results**

spectrum	=	spectrum.fits	image/fits	↑	↓	×
Desc. Description						
spectrum_preview	=	spectrum.png	image/png	↑	↓	×
Desc. Description						
<span>Add result</span> <span>Remove all results</span>						

List of possible results with their name and content type. A default name can be provided. Note that a result can refer to a parameter (if it has the same name), e.g. the name of an output file generated by the script.

**Parameters**

configfile	=	make_spectra.cfg	Req.? <input type="checkbox"/>	xs:string	↑	↓	×
Desc. Configuration file (generated by the script)							
Options List of possible choices (comma-separated values)							
Attr. unit=... ucd=... utype=... min=... max=...							
RA	=	329.7169379	Req.? <input checked="" type="checkbox"/>	xs:double	↑	↓	×
Desc. Target Blight Ascension							

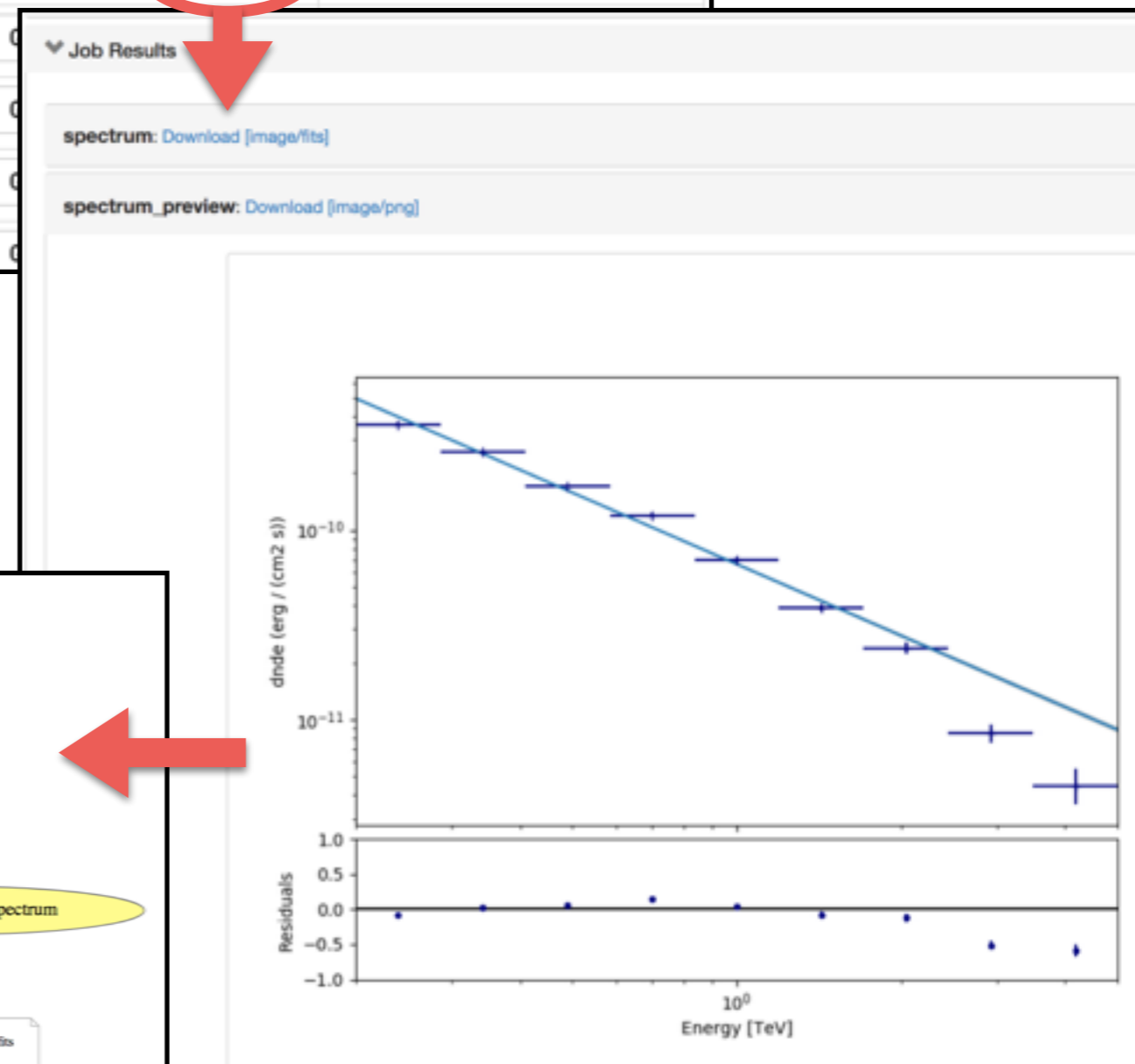
List of parameters, with name, default value, type and description. Specify if the parameter is required by checking the box (if not, the parameters won't be shown by the client and the default value will always be used). A list of options can be specified (comma-separated values).

# Client - Job list

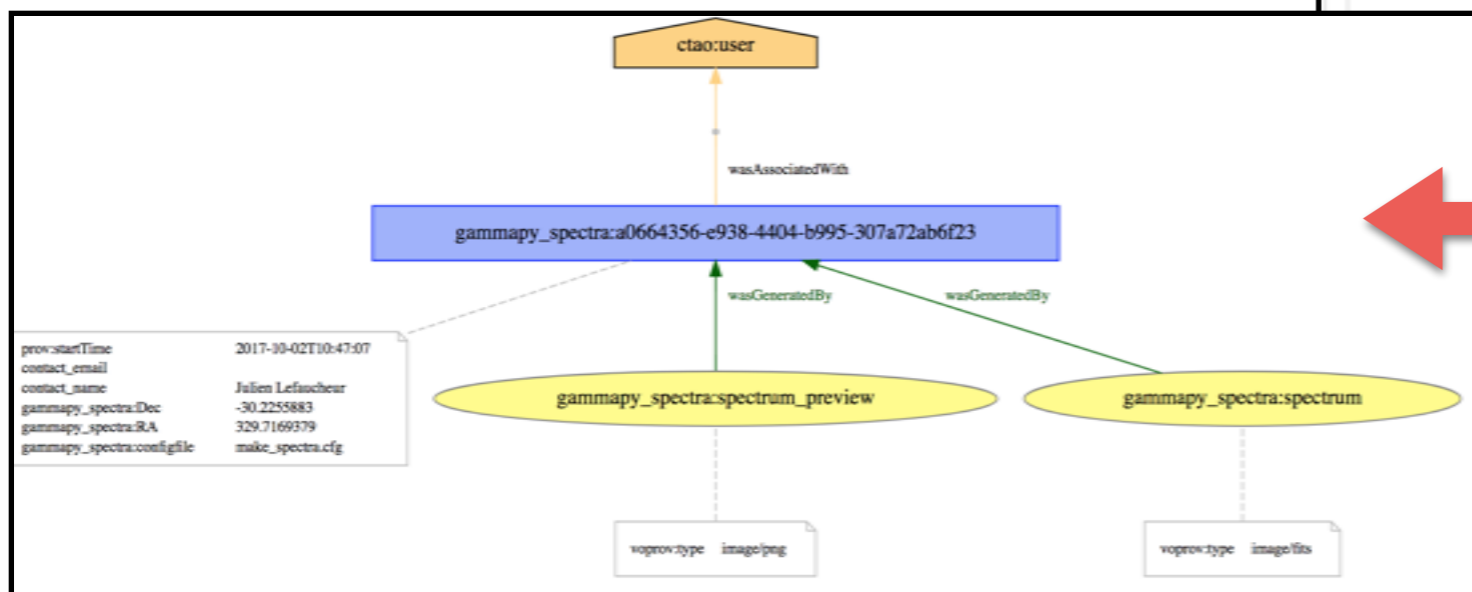
OPUS Job Definition Job List Signed in as user

Job List for **gammapy\_spectra** Refresh Job List Create Test Job Create New Job

Type	Start Time	Destruction Time	Phase	Details	Control
<b>gammapy_spectra</b>	2017-10-02 10:47:07	2017-11-01 10:47:05	COMPLETED	Properties Parameter <b>Results</b>	Start Abort Delete
<b>gammapy_spectra</b>		2017-11-01 10:47:03	PENDING	Properties	
<b>gammapy_spectra</b>	2017-09-29 15:07:52	2017-10-29 15:07:51	COMPLETED	Properties	
<b>gammapy_spectra</b>	2017-09-29 14:55:10	2017-10-29 14:55:09	ABORTED	Properties	
<b>gammapy_spectra</b>	2017-09-29 14:21:20	2017-10-29 14:21:19	COMPLETED	Properties	



## Tracking of Provenance informations



# Serializations - W3C PROV formats

```
<prov:document xmlns:ctadata="ivo://vopdc.obspm/cta#" xmlns:ctajob
  <prov:activity prov:id="ctajobs:ctbin">
    <prov:startTime> 2016-03-13T23:44:46 </prov:startTime>
    <prov:endTime> 2016-03-13T23:44:56 </prov:endTime>
  </prov:activity>
  <prov:agent prov:id="cta:consortium">
    <prov:type xsi:type="xsd:string"> Organization </prov:type>
  </prov:agent>
  <prov:wasAssociatedWith>
    <prov:activity prov:ref="ctajobs:ctbin" />
    <prov:agent prov:ref="cta:consortium" />
  </prov:wasAssociatedWith>
  <prov:entity prov:id="uwsdata:parameters/inobs" />
  <prov:used>
    <prov:activity prov:ref="ctajobs:ctbin" />
    <prov:entity prov:ref="uwsdata:parameters/inobs" />
  </prov:used>
  <prov:entity prov:id="uwsdata:results/outcube" />
  <prov:wasGeneratedBy>
    <prov:entity prov:ref="uwsdata:results/outcube" />
    <prov:activity prov:ref="ctajobs:ctbin" />
  </prov:wasGeneratedBy>
  <prov:wasDerivedFrom>
    <prov:generatedEntity prov:ref="uwsdata:results/outcube" />
    <prov:usedEntity prov:ref="uwsdata:parameters/inobs" />
  </prov:wasDerivedFrom>
  <prov:entity prov:id="uwsdata:results/logfile" />
  <prov:wasGeneratedBy>
    <prov:entity prov:ref="uwsdata:results/logfile" />
    <prov:activity prov:ref="ctajobs:ctbin" />
  </prov:wasGeneratedBy>
  <prov:wasDerivedFrom>
    <prov:generatedEntity prov:ref="uwsdata:results/logfile" />
    <prov:usedEntity prov:ref="uwsdata:parameters/inobs" />
  </prov:wasDerivedFrom>
</prov:document>
```

```
{
  - wasAssociatedWith: {
    - _:id1: {
      prov:agent: "cta:consortium",
      prov:activity: "cta:anactools_v1.1"
    }
  },
  - agent: {
    - cta:consortium: {
      prov:type: "Organization"
    }
  },
  - entity: {
    uwsdata:results/fit_results: { },
    uwsdata:results/configfile: { },
    uwsdata:results/butterfly: { },
    uwsdata:results/spectrum_plot: { },
    uwsdata:results/spectrum: { }
  },
  - prefix: {
    uwsdata: "https://voparis-uws-test.obspm.fr/real",
    cta: "http://www.cta-observatory.org#",
    voprov: "http://www.ivoa.net/ns/voprov#"
  },
  - activity: {
    - cta:anactools_v1.1: {
      prov:startTime: "2016-04-07T00:26:00",
      prov:endTime: "2016-04-07T00:27:15"
    }
  },
  - wasGeneratedBy: {
    - _:id5: {
      prov:entity: "uwsdata:results/butterfly",
      prov:activity: "cta:anactools_v1.1"
    },
    - _:id4: {
      prov:entity: "uwsdata:results/fit_results",
      prov:activity: "cta:anactools_v1.1"
    }
  },
}
```