UWS v1.1

Propositions

Jonathan Normand, Pierre Le Sidaner
VO-Paris Data Centre
Jean-Christophe Malapert
CNES

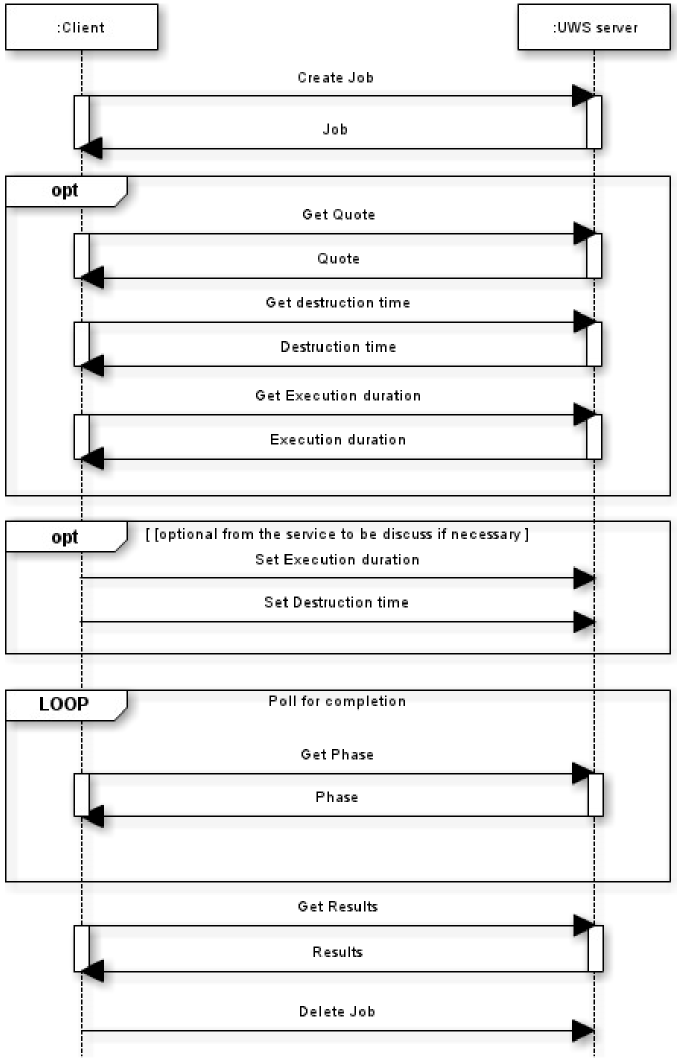
IVOA InterOp - Pune, Octobre 2011

UWS 1.1

- **Goal: make it simple and easy to implement and use**
  - Describe all necessary resources and avoid multiple occurrences
  - Define all supported operations and return code
  - Use REST formalism
  - Exclude ambiguity: only one way to make an action
- **Add**
  - Pagination mechanism
  - Upload file capability

UWS 1.1: Sequence diagram

Typical calling sequence



UWS 1.1: way to execute a job

- **Job execution: create, set and start in one time**
  - HTTP POST[p1,p2...] on /{jobs}
    - Set parameters by value or reference (ie. URI)
      - Content-Type of request = application/x-www-form-urlencoded
    - Set parameters by uploading file
      - Content-Type of request = multipart/form-data
    - Mix
      - Content-Type of request = multipart/form-data
  - HTTP response code
    - 201, 400, 415, 500
- **=> No multiple steps to send job for execution**

UWS 1.1: way to get results

- **Poll for completion (or terminated state)**
  - Like as v1.0
- **Get results (if any)**
  - Like as v1.0
- **Delete job from client**
  - Only one method: HTTP DELETE on /{jobs}/{job-id}
    - Tunneling API exists for server application to handle DELETE method.

UWS 1.1: Completion Time

- **Quote and Execution duration**
  - Quote
    - Represents time when the job is likely to complete. Difficult to predict. Accept “don't know” value
  - Execution time
    - Represents computation time allowed. Accept unlimited time
- **But what is useful for the user ?**
  - The time when the results are available / job is likely to completed
- **Propose one time : Completion time (or whatever)**
  - Represents the time when the job is to likely completed
  - Absolute time. ISO8601
  - Must be provided by the service or at least an estimation
  - Remove Quote and Execution duration objects

Useless capabilities

- **Remove capability to set Completion time and Destruction time**
  - Completion time: usually users don't know architecture (CPU...), how jobs are managed (batch queue, scheduler or not) and if the execution time of job is parameters dependent.
  - Plus: usually job management system does not allow user to increase execution duration previously set.
    - No interest to set completion time
  - There is no clear interest to keep the possibility to set Destruction time. Service sets this time and user has to get results before it.

Useless phase PENDING

- **V1.0 mechanism**
  - What is the interest to create job (put in PENDING), set parameters and start (PHASE=RUN) it in 3 steps ?
    - More steps but no more functionality
  - Why create a job if you don't want to execute it ?
- **V1.1 mechanism**
  - Job goes directly to QUEUED or EXECUTING (see. Previous slide)
  - Only one step !
- **PENDING should be interested in this use case**
  - Job waiting for parameters can be useful in workflow BUT it is managed by the workflow management system
- **=> Remove PENDING phase**

Useless user action

- **V1.0**
  - Client can abort job
  - But no guaranty on the validity of the available results
  - What is the purpose of this action ?
- **Possible use case**
  - Should be interesting if user has intermediate result and is able to restart the job from it. BUT this could be done in a 2 steps service without the need to abort.
- **=> Remove the user possibility to abort a job**

UWS 1.1: Pagination

Pagination mechanism

- To get objects with lot of children
  - HTTP GET on /{jobs}?start=<first>&extend=<amount>
    - First: number of the first job
    - Amount: amount of job to retreive
  - Need to know the amount of job in the JobList
    - Use HTTP HEAD method on /{jobs} with a custom metadata in the header
    - header(JobAmount: x) for example
  - By default HTTP GET on /{jobs} returns the first page

UWS 1.1: Authentication

Authentication mechanism

- Use HTTP protocol
  - RFC 2616 – HTTP/1.1
  - RFC 2617 – HTTP Authentication: Basic and Digest Access Authentication
  - RFC 1321 – The MD5 Message-Digest Algorithm
- Add the following response on all resource
  - HTTP 401 Unauthorized
  - http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html#sec10.4.2

UWS 1.1: JDL

How to set parameters

- For services where the JDL consists of a list of name/value pairs (typical of the standard IVOA "simple" access protocols), then these would naturally be expressed in the parameter list.
- For services where the JDL consists in a document with its own syntax (for instance an XML document with a specific schema, JSON file...), then there would be a single <uws:parameter> element where the content was the URL to that document.
  - Instead of writing the whole document as value of <parameter> element we propose to indicate the URL of that document
  - <uws:parameter id="jdl" byReference="true"> http://uws.example.org/jobs/job1/parameters/jdl </uws:parameter>