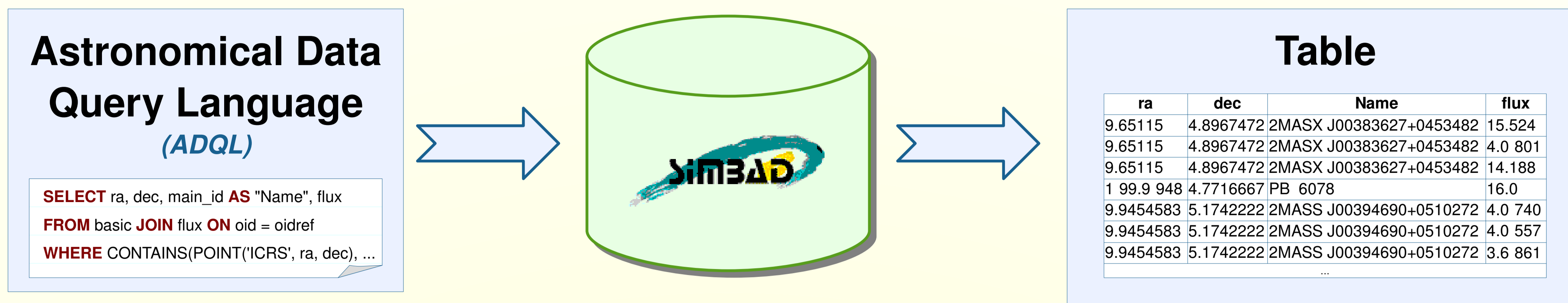


Grégory Mantelet <sup>[1]</sup>, Laurent Michel <sup>[2]</sup>, Marc Wenger <sup>[1]</sup>

[1] CDS, [2] Observatoire astronomique de Strasbourg



### TAP: IVOA Protocol

TAP or Table Access Protocol is an IVOA Web-service protocol that gives access to collections of tabular data referred to collectively as a tableset. TAP services accept queries posed against the tableset available via the service and return the query response as another table, according to the relational model. Queries may be submitted using various query languages and may execute synchronously or asynchronously. The output of a TAP query is a table, normally returned as a VOTable. Support for VOTable output is mandatory; all other formats are optional.

Default query language: ADQL  
Default output format: VOTable

<http://www.ivoa.net/Documents/TAP/>

### ADQL: SQL-Like Language

The Astronomical Data Query Language (ADQL) is the language used by the IVOA to submit astronomy queries to VO services. In addition to the default functionalities of SQL, ADQL provides functions to select objects contained in a given sky region or which intersect with another region.

Some geometrical functions:

- POINT('ICRS', ra, dec)
- CIRCLE('ICRS', 10, 5, 1)
- CONTAINS(<REGION>, <REGION>) = 1

Example – Cone search:  
WHERE CONTAINS(POINT('ICRS', ra, dec), CIRCLE('ICRS', 10, 5, 1)) = 1;

<http://www.ivoa.net/Documents/latest/ADQL.html>

### Simbad-TAP Implementation

#### How did we implement TAP on Simbad ?

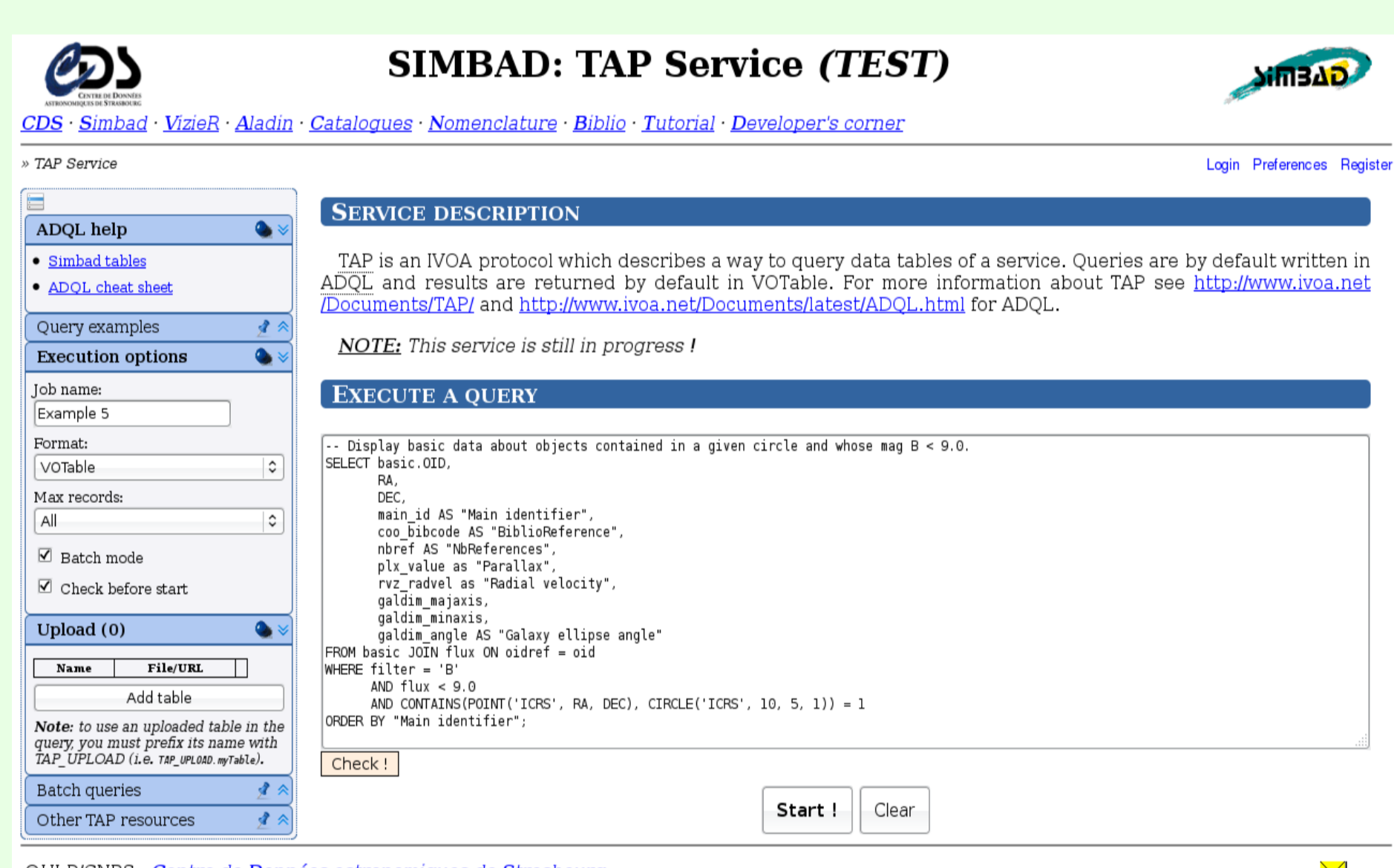
- Update the Simbad database**
  - Create and fill the schema TAP\_SCHEMA which must describe all metadata about exposed tables and columns
  - Add PgSphere into PostgreSQL
  - Install PgSphere
  - Create an SPOINT column from RA and DEC
  - Index columns with a geographic information (particularly SPOINT columns)
- Implement TAP in the Simbad server by using the CDS TAP Library**
  - Configure the database access (interface DBConnection)
  - Provide the list of available metadata (class TAPMetadata)
  - Create one formatter for each output format (interface OutputFormat)
  - Describe the service (interface ServiceConnection)
  - Write a servlet which forwards all requests to the library

### Simbad TAP

#### Output formats

VOTable, JSON, CSV, TSV, Text

#### Web interface



**Also usable with:**

- TopCAT <http://www.star.bris.ac.uk/~mbt/topcat/>
- TapHandle <http://saada.u-strasbg.fr/taphandle/>

### Tables Mapping

Tables exposed by Simbad-TAP are not the true database tables. Indeed a mapping is processed between Simbad tables and Simbad-TAP tables. Thus possible evolutions of the Simbad database are totally independant from its TAP service.

This mapping is done automatically by the CDS ADQL Library, if the metadata of exposed tables are provided through the following interfaces:

```

«DBTable»
+ getADQLName(): String
+ getDBName(): String
+ getADQLSchemaName(): String
+ getDBSchemaName(): String
+ getADQLCatalogName(): String
+ getDBCatalogName(): String
+ getColumn(String, boolean): DBColumn
+ copy(String, String): DBTable

«DBColumn»
+ getADQLName(): String
+ getDBName(): String
+ getTable(): DBTable
+ copy(String, String, DBTable): DBColumn

```

### Upload (optional)

A TAP service may have an UPLOAD capability which allows the user to upload his own tables on the server and use them in an ADQL query as the Simbad-TAP tables. Uploaded tables must stay on the server only during the query execution. To reference such tables in ADQL, their name must be prefixed by: TAP\_UPLOAD (ex: TAP\_UPLOAD.myTable).

In Simbad-TAP, uploading a table (VOTable) means creating a table under the TAP\_UPLOAD schema of the Simbad database. When the service detects the end of the query all tables uploaded for this query are dropped from the database.

### Geometrical Functions

The CDS ADQL library is able to translate ADQL into SQL except for the geometrical functions. Indeed no SQL functions correspond to ADQL geometrical functions. So their execution is totally dependant on the TAP implementation.

Simbad objects are stored in a PostgreSQL database. At least 2 plugins for PostgreSQL are designed to manage geographical information: PgSphere and Q3C.

Simbad-TAP uses PgSphere. Consequently, all ADQL geometrical functions are translated in a SQL compatible with this plugin.

Example – Search cone:  
CONTAINS(POINT('ICRS', ra, dec), CIRCLE('ICRS', 10, 5, 1)) = 1;  
=<=>  
(coord @ scircle(spointradians(10),radians(5)),radians(1))) = '1'

### Soon available

Simbad-TAP will be available by the end of the year. You will be able to access it through the Simbad website.

### Questions ?

[cds-question@astro.unistra.fr](mailto:cds-question@astro.unistra.fr)

### Coordinate System

As for the execution of geometrical functions, coordinate system transformations depend completely on the TAP implementation.

Currently, Simbad-TAP does not yet process these transformations. ICRS is the only allowed coordinate system.

## CDS Libraries

The protocol TAP is dependent on two IVOA standards. The first one, ADQL, is the default and mandatory language used to query a TAP service. The second standard is UWS and is used to execute ADQL queries in an asynchronous manner. Thus, when you want to implement the TAP protocol, you must also implement UWS and ADQL.

The CDS provides 3 libraries to implement as easily and as quickly as possible these IVOA standards. Obviously, the TAP library includes the ADQL and UWS libraries, which can be used independantly in other applications or services (i.e. the UWS library is used for the CDS crossmatch service).

TAP Library	ADQL Library	UWS Library
<ul style="list-style-type: none"> <li>execute ADQL queries</li> <li>describe table metadata</li> <li>let uploading tables</li> </ul>	<ul style="list-style-type: none"> <li>parse ADQL</li> <li>manipulate the ADQL tree</li> <li>translate into SQL</li> </ul>	<ul style="list-style-type: none"> <li>execute jobs in background</li> <li>manage execution queue</li> <li>list and summarize jobs</li> </ul>

<http://cds.u-strasbg.fr/resources/doku.php>

TAP Features		Managed ?
TAP	languages	ADQL ✓ PQL ✓
	query executions	synchronous ✓
		asynchronous (UWS) ✓
		availability ✓
	resources	capabilities (with TAPRegExt) ✓
		tables ✓
	parameters	request=doQuery ✓
		request=getCapability ✓
		version ✓
		query ✓
		format ✓
		maxRec ✓
		runId ✓
	ADQL	upload (inline) ✓
		upload (http) ✓
TAP_UPLOAD (db schema) metadata ✓		
TAP_SCHEMA (db schema) ✓		
ADQL	parse	PostgreSQL+PgSphere ✓ other DBMs ✓ others ✓
	Execute	coordinate system ✓ check with DB ✓

**Legend**

- ✓ Fully managed
- ✓ Specific extension required
- ✗ Not yet managed
- ✓ No generic implementation possible