Observation Locator TAP - ObsLocTAP

Jesús Salgado[1], Aitor Ibarra[1], Karl Foster[5], Richard Saxton[2], Jan-Uwe Ness[4], Erik Kuulkers[4], Carlos Gabriel[4], Bruno Merin[4], Peter Kretschmar[4],Matthias Ehle[4], Emilio Salazar[3], Celia Sánchez[3]

1 Quasar for ESA

2 TPZ-VEGA for ESA

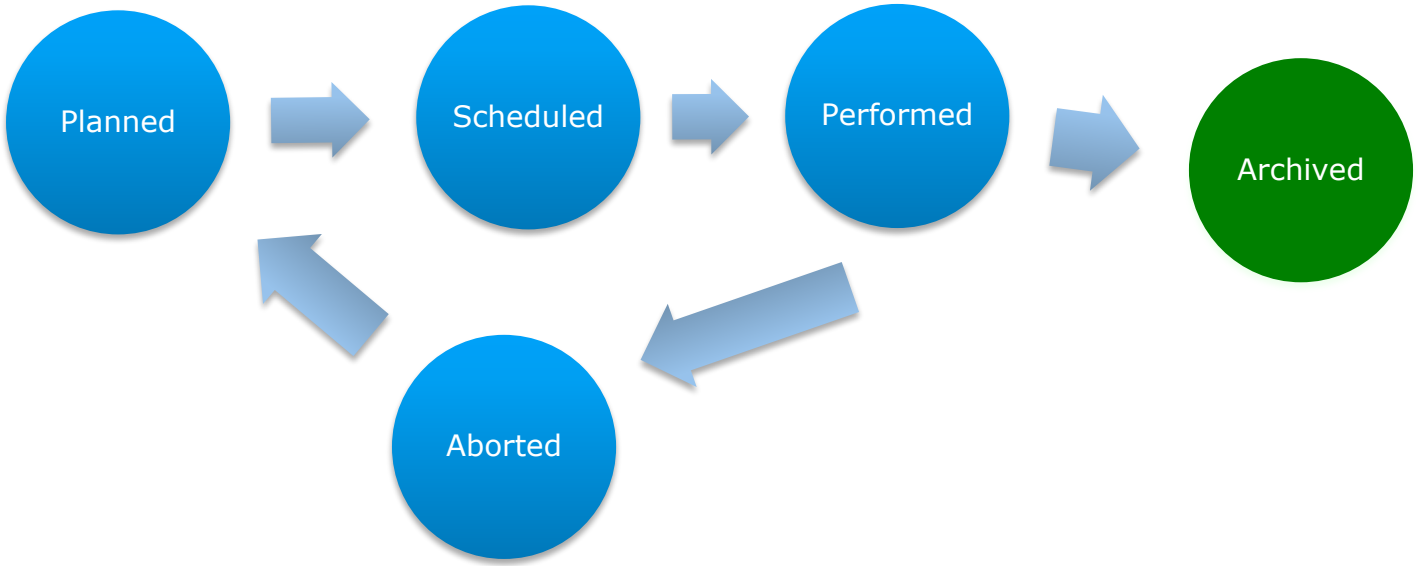
3 ATG for ESA

4 ESA

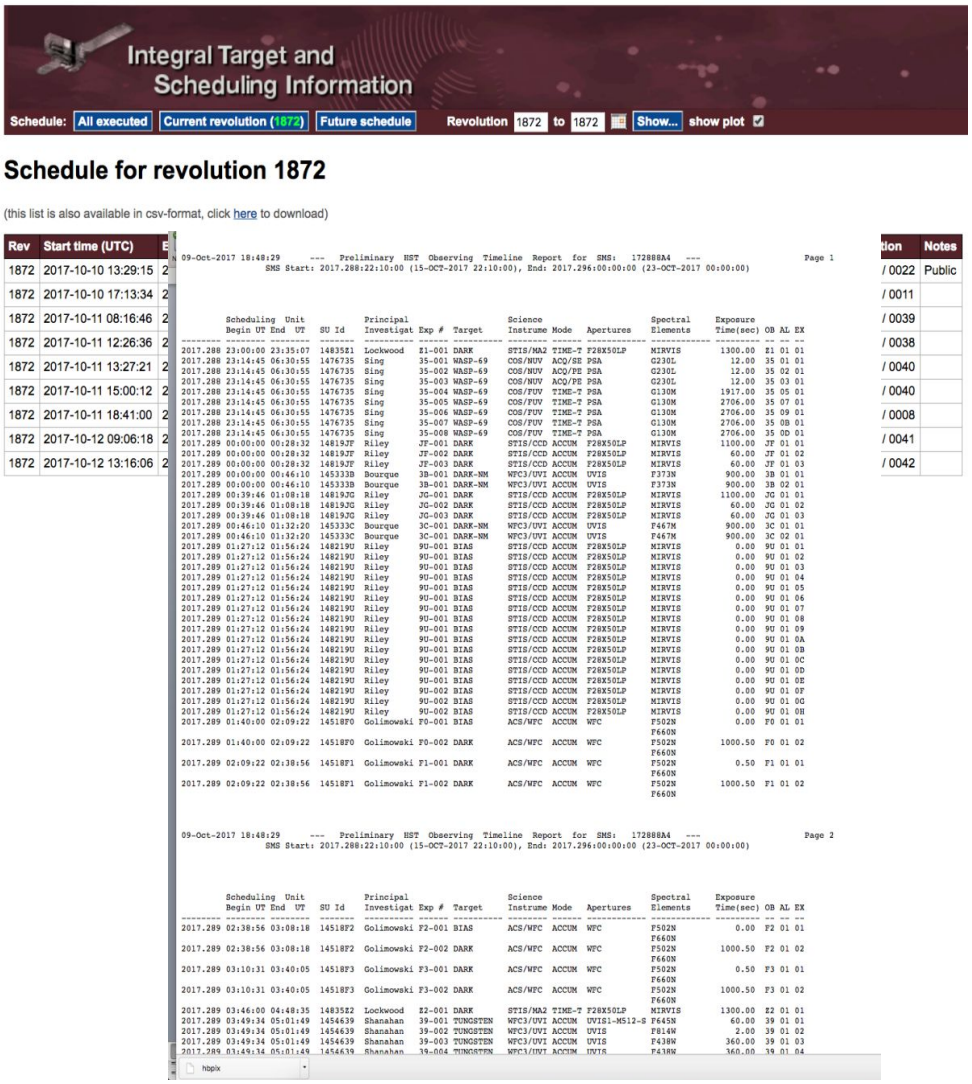
5 NuSTAR

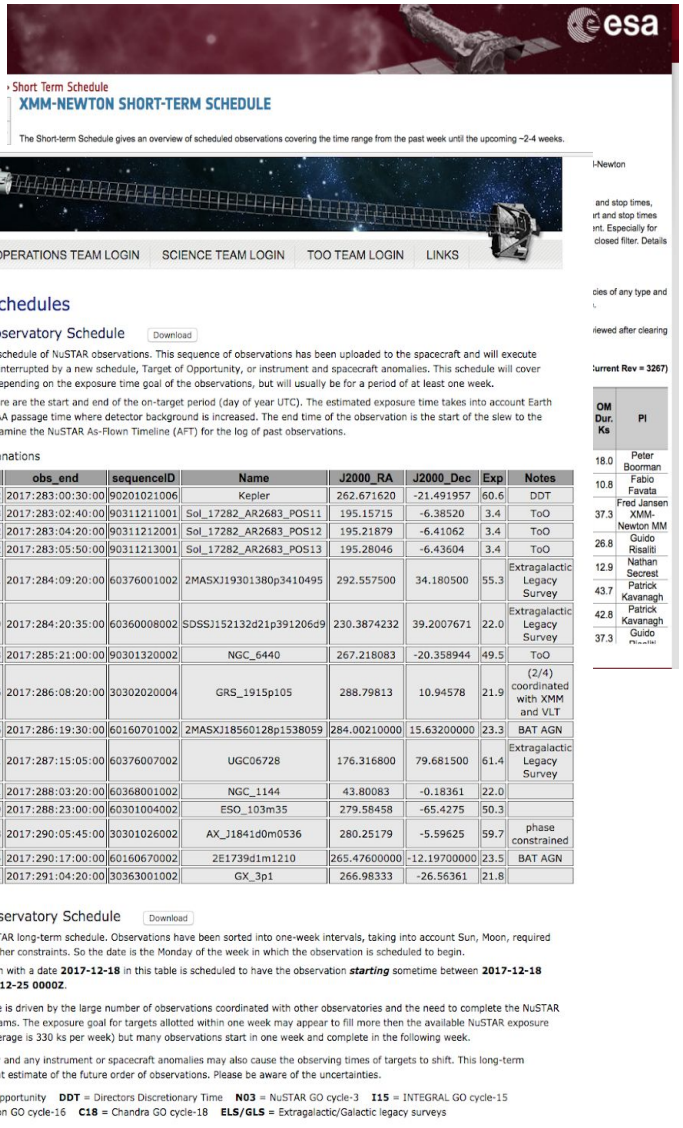
Use Case

Observations Life cycle



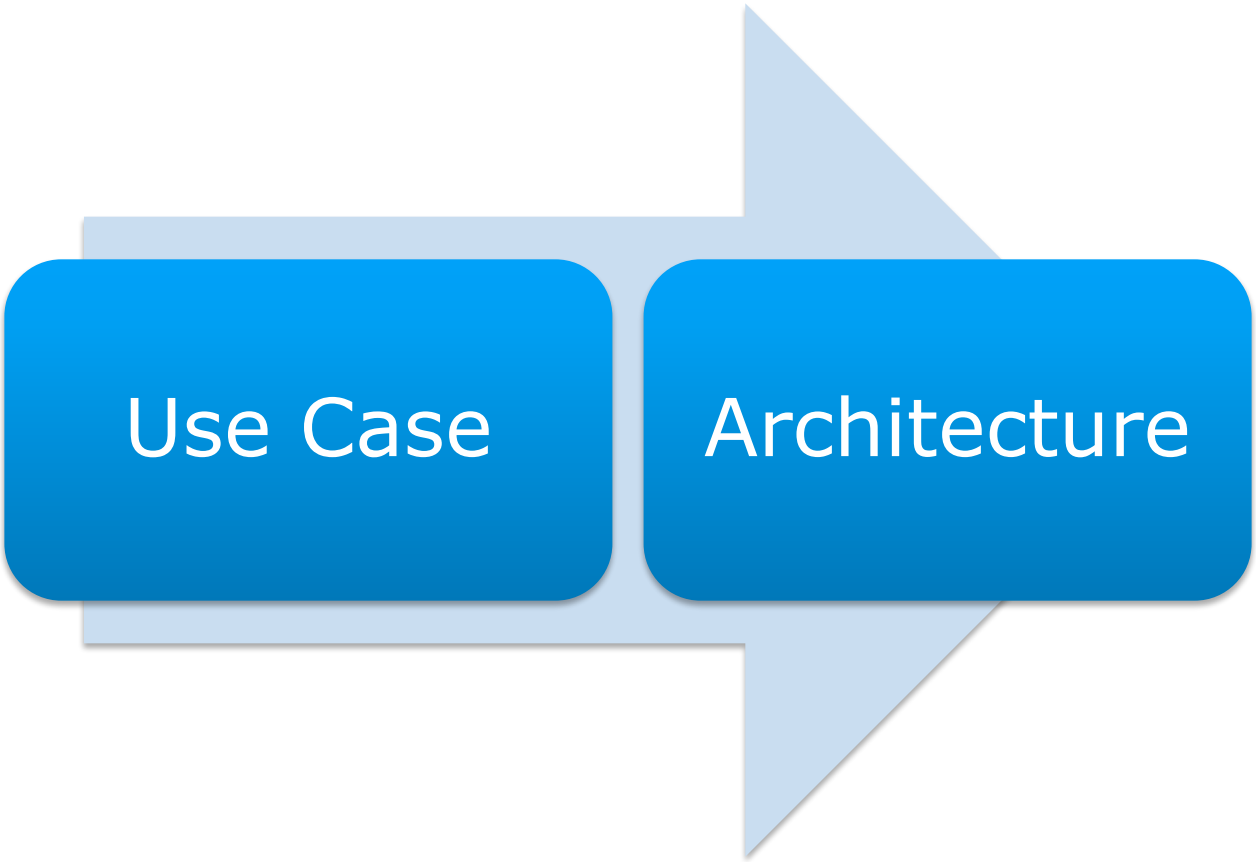
Planned Observations Services



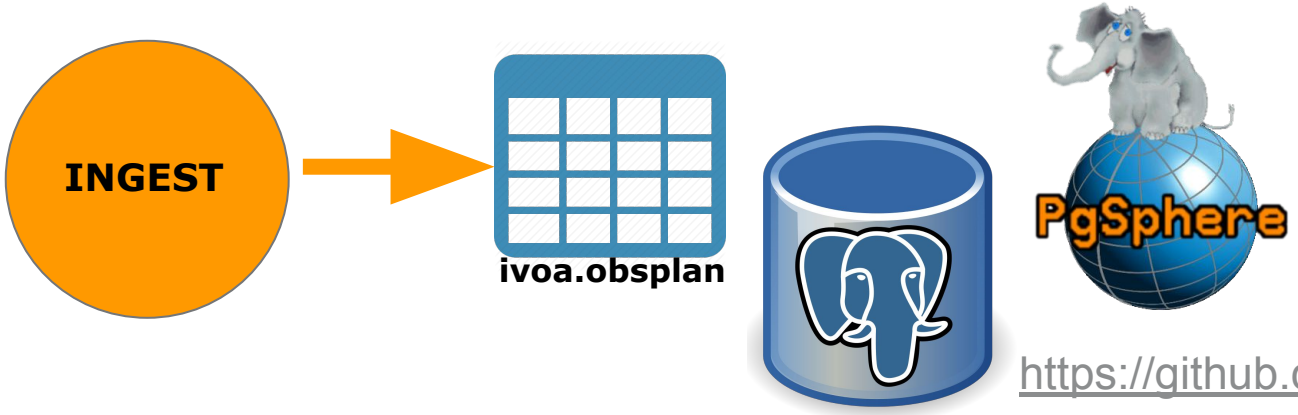


Goals of this tutorial

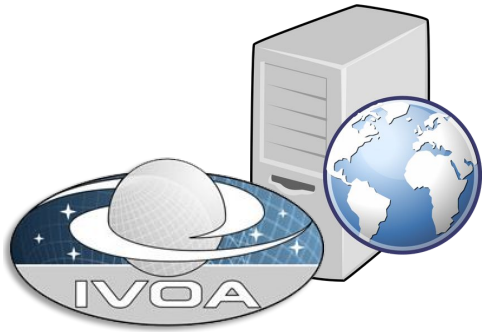
- Understand the role of all elements needed to implement an ObsLocTAP server
- Explain how to follow the implementation guide: https://www.cosmos.esa.int/web/vovisobs_protocols/implementation-guides
- Identify difficult steps into the implementation
- Get feedback from the different partners to understand the status and possible support



ObsLocTAP architecture



https://github.com/akorotkov/pgsphere



http://cdsportal.u-strasbg.fr/taptuto/

Why TAP?

- **TAP (Tabular Access Protocol)** is a standard defined by the IVOA (International Virtual Observatory Alliance) to query astronomical databases
- It allows synchronous and asynchronous queries
- Used for many astronomical data providers (ESA, CDS, Heasarc, LSST...)
- A lot of client applications, including User Interfaces (e.g. TopCat) and command line interfaces (e.g. python astropy/pyvo)
- It is powerful and allows users (and applications) to create many complex exploration queries
- It allows the use of **ADQL**

Why ADQL?

- **ADQL** (Astronomical Data Query Language) is a SQL based query language designed for Astronomical Data Bases
  - No limitations on WHERE conditions
  - No restrictions on columns for SELECT
  - Spatial support based on coordinates
    - CONTAINS/INTERSECTS
    - CIRCLE, POLYGON, POINT
  - Enables temporary tables uploaded from users (persistent tables for TAP+)
  - Due to security restrictions, it does not support procedures and functions to modify alter database contents

ADQL Examples

Gaia Archive

```
SELECT DISTANCE(
  POINT('ICRS', ra, dec),
  POINT('ICRS', 266.41683, -29.00781)) AS dist, *
FROM gaiadr2.gaia_source
WHERE 1=CONTAINS(
  POINT('ICRS', ra, dec),
  CIRCLE('ICRS', 266.41683, -29.00781, 0.08333333))
ORDER BY dist ASC
```

ADQL Examples (II)

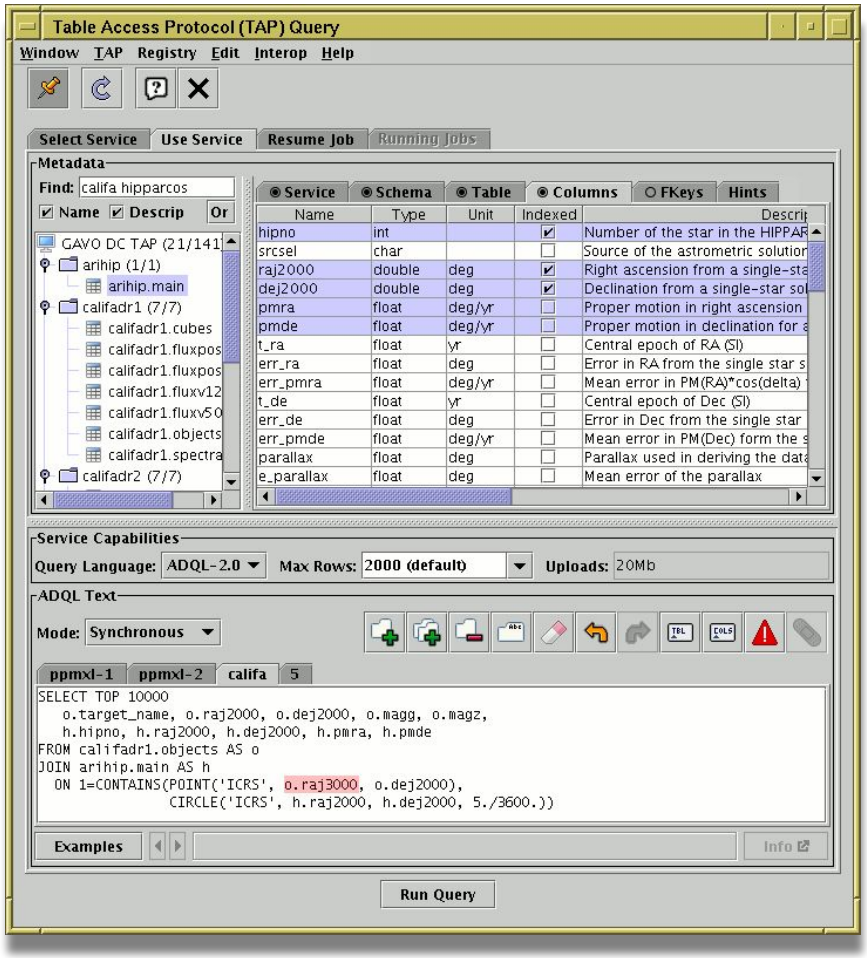
ObsLocTAP Query

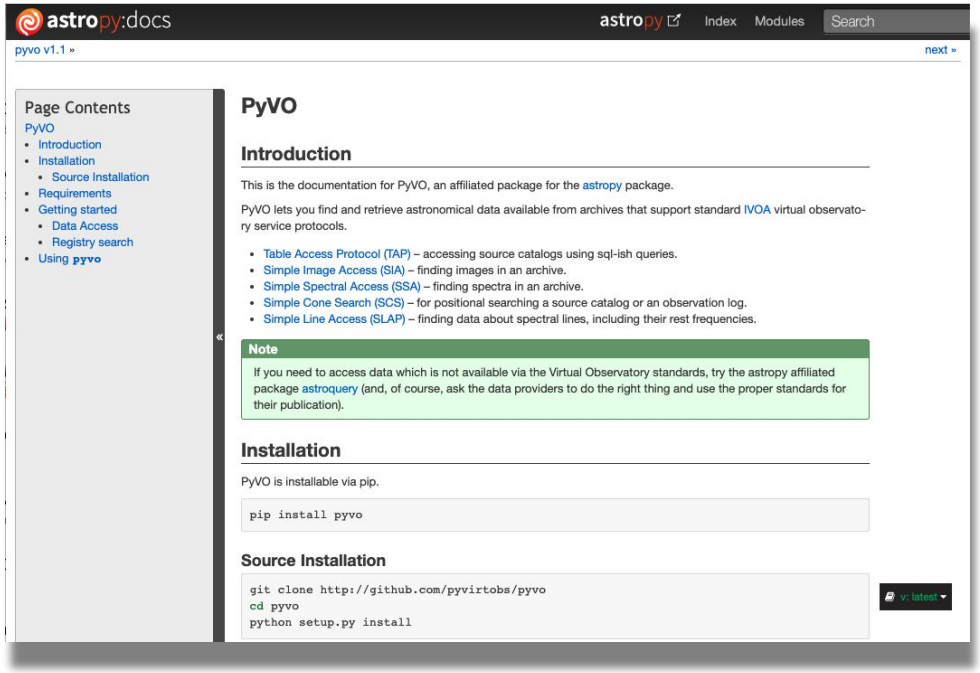
```
SELECT * FROM
ivoa.obsplan
WHERE
t_planning > 58500 AND t_max < 58502
AND 1=INTERSECTS(s_region, CIRCLE('ICRS', 114.8251,
1.6179, 0.016666 ))
```

where we are checking whether there are any newly planned observations from 58500 (17/01/2019) on the next two days (58502 or 19/01/2019) around the target PKS 0736+017, with a radius of 1 arcmin

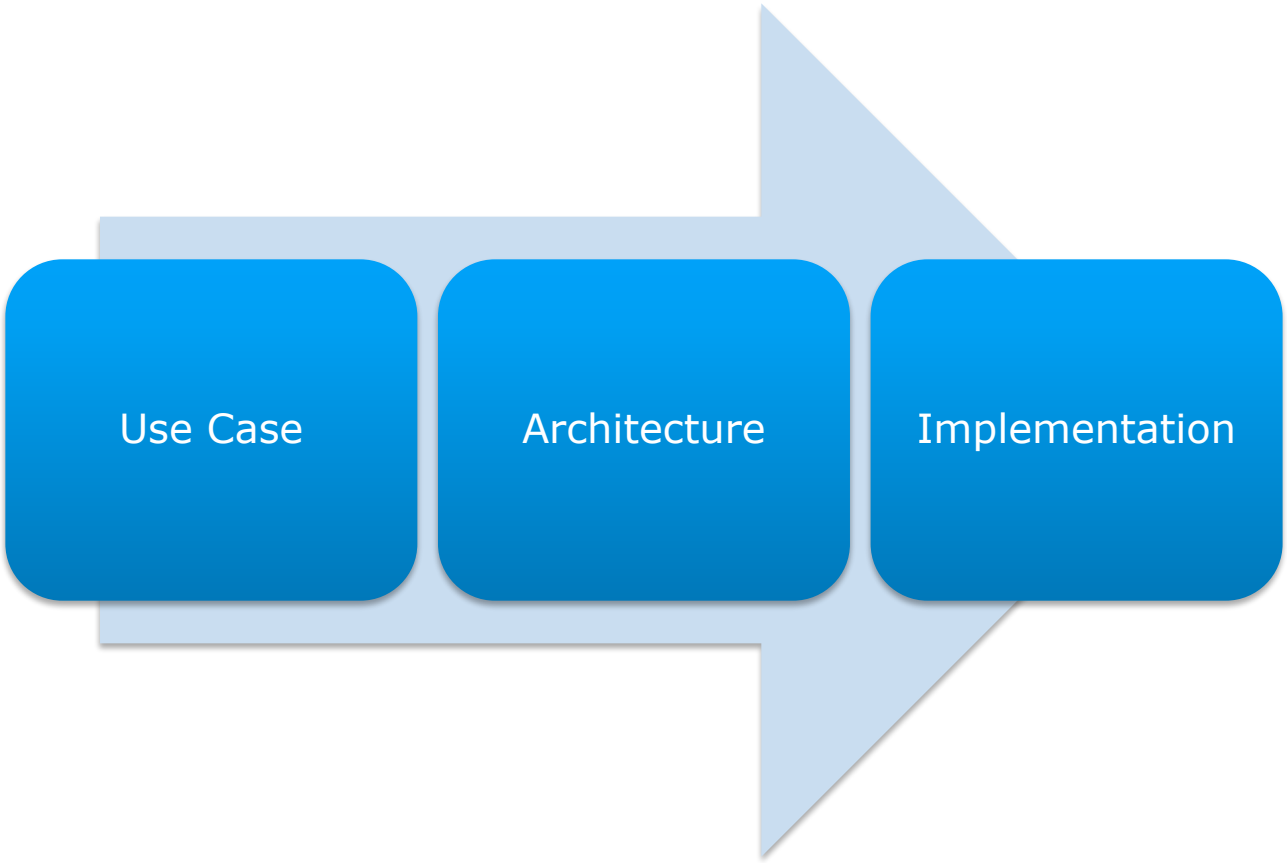
TAP Clients

http://www.star.bris.ac.uk/~mbt/topcat/sun253/TapTableLoadDialog.html



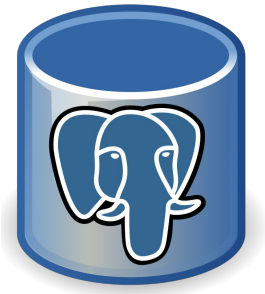


https://pyvo.readthedocs.io/en/latest/



ObsLocTAP implementation: Database

- **Install a PostgreSQL DB:** Recommended 9.6
  - Commonly used in Astronomy
  - Geometrical indexing support
  - IVOA Toolkits compatible with PosgreSQL



https://www.postgresql.org/download/

- **Install pgSphere:** PostgreSQL modules which adds spherical data types.
  - containing, overlapping, and other operators
  - point, circle, polygon
  - indexing of spherical data types

https://pgsphere.github.io/

- There are some shortcuts for some operating systems like:

  https://zoomadmin.com/HowToInstall/UbuntuPackage/postgresql-pgsphere

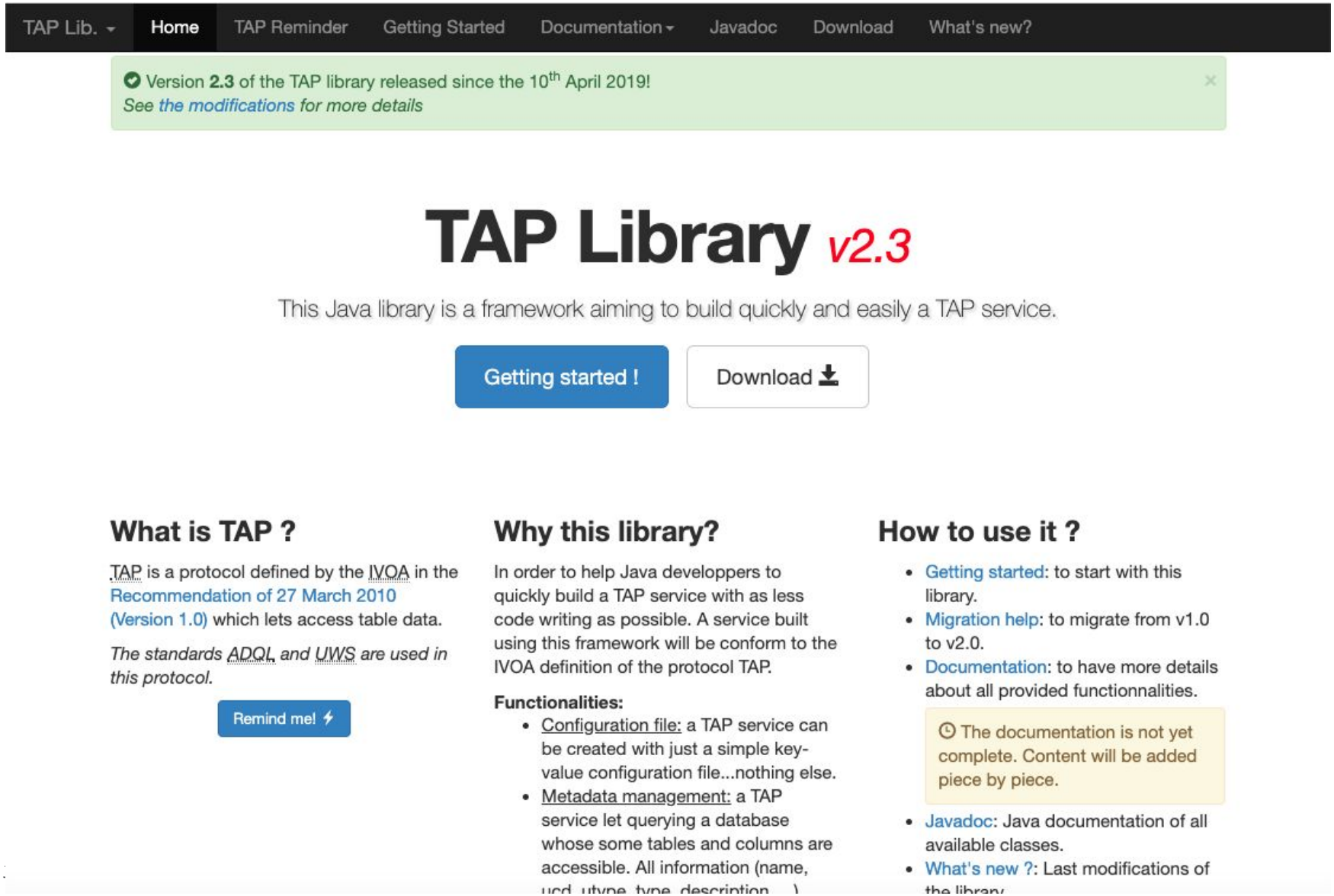
  but it is usually better to install both packages sequentially

ObsLocTAP implementation: TAP

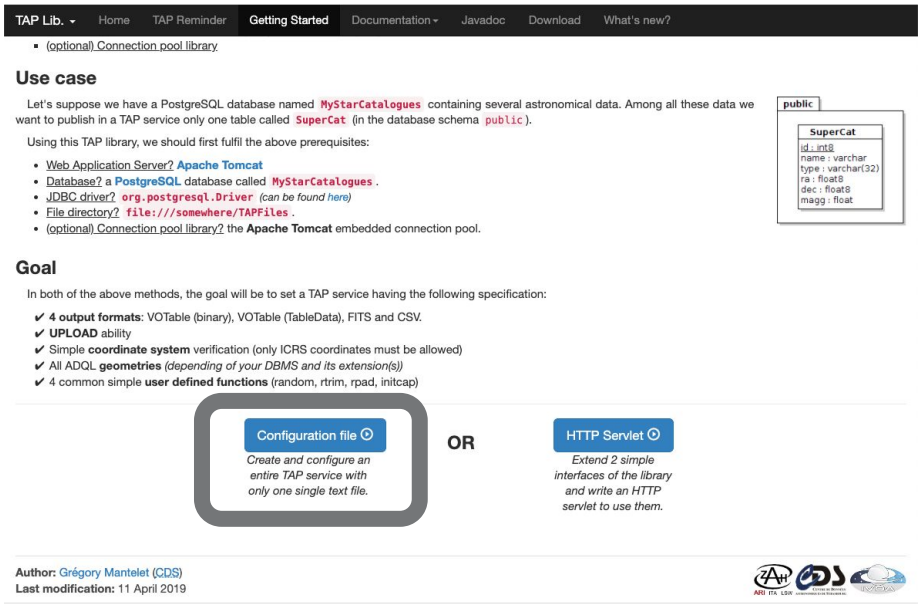
- IVOA offers different toolkits
  - DACHS from GAVO:
    - http://soft.g-vo.org/dachs https://dachs-doc.readthedocs.io/tutorial.html
  - SAADA:
    - http://saada.unistra.fr/saada/
  - TAPTuto:
    - http://cdsportal.u-strasbg.fr/taptuto/

http://cdsportal.u-strasbg.fr/taptuto/

- Very well documented
- Clear instructions
- Always updated to latest version (e.g. ADQL)
- Note: It makes use of a Tomcat web server (if not available, it needs to be installed too)


TAP Lib.
Home
TAP Reminder
Getting Started
Documentation
Javadoc
Download
What's new?
Version 2.3 of the TAP library released since the 10th April 2019!
See the modifications for more details
TAP Library v2.3
This Java library is a framework aiming to build quickly and easily a TAP service.
Getting started !
Download
What is TAP ?
TAP is a protocol defined by the IVOA in the Recommendation of 27 March 2010 (Version 1.0) which lets access table data.
The standards ADQL and UWS are used in this protocol.
Remind me!
Why this library?
In order to help Java developpers to quickly build a TAP service with as less code writing as possible. A service built using this framework will be conform to the IVOA definition of the protocol TAP.
Functionalities:
Configuration file: a TAP service can be created with just a simple key-value configuration file...nothing else.
Metadata management: a TAP service let querying a database whose some tables and columns are accessible. All information (name,
How to use it ?
Getting started: to start with this library.
Migration help: to migrate from v1.0 to v2.0.
Documentation: to have more details about all provided functionnalities.
The documentation is not yet complete. Content will be added piece by piece.
Javadoc: Java documentation of all available classes.
What's new ?: Last modifications of

ObsLocTAP implementation: TAP trick I



- **To configure the TAP service the easiest way is to use the configuration file option**
  - No code to be written
  - Change MyTAP servlet-name to your project name

```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd" version="3.0">
    <servlet>
        <display-name>tap</display-name>
        <servlet-name>MyTAP</servlet-name>
        <servlet-class>tap.config.ConfigurableTAPServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>MyTAP</servlet-name>
        <url-pattern>/tap/*</url-pattern>
    </servlet-mapping>
</web-app>
```

ObsLocTAP implementation: TAP trick II

- **Connection to DB can be done by properties or JNDI**
  - Through properties is very simple. JNDI for better performance
  - Change jdbc_url schema to your scheme (probably ivoa)

Method 1: JDBC parameters

Just 1 parameter to select the JDBC method and 4 additional parameters to configure it are required here, nothing else:

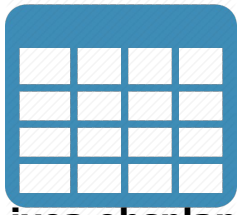
```
database_access = jdbc
jdbc_driver = org.postgresql.Driver
jdbc_url = jdbc:postgresql:MyStarCatalogues
db_username = tapuser
db_password = *****
```

- **DB schema `TAP_SCHEMA`**
  - TAP_SCHEMA created at DB. Scripts provided by the instructions
- Use `sql_translator = pgsphere`
  - The translator (pgsphere instead of postgresql) is one of the few differences of the instructions

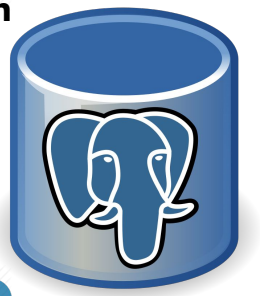
ObsLocTAP Data Model

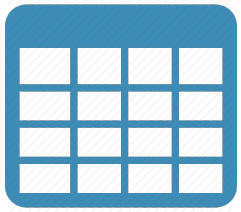
INGEST

- obsplan table needs to be created into DB
- Table should be populated



ivoa.obsplan





TAP_SCHEMA

- Defines with tables and columns are published into the TAP service
- Can be used for aliases
- Security layer. Not all the tables and columns are published

TAP_SCHEMA

- Create a basic/plain TAP_SCHEMA using TAPTUTO
    - **TAP_SCHEMA.schemas**: One row per schema where TAP tables are located
    - **TAP_SCHEMA.tables:** One row per table published on TAP
    - **TAP_SCHEMA.columns:** One row per column published on TAP
- A new schema (ivoa) a new table (obsplan) and some columns need to be added to the ObsLocTAP service TAP_SCHEMA
- These are described and an example of the SQL inserts could be found at the implementation guide:
    - https://www.cosmos.esa.int/web/vovisobs_protocols/implementation-guides

ivoa.obsplan creation

- Next step is the obsplan DB creation
- A script can be found at:

  https://www.cosmos.esa.int/web/vovisobs_protocols/implementation-guides

  - CREATE EXTENSION pg_sphere;
  - Pgsphere will be needed to create the s_region field. Two possible types:
    - Circle
    - Polygon

  - Unfortunately, there is no way to express a mixture of circles and polygons in the same row
  - Ingestion can be done talking directly with the database (e.g. using psql)

INSERT example

```
INSERT INTO ivoa.obsplan
(t_planning,target_name,obs_id,obs_collection,s_ra,s_dec,s_fov,s_region,s_resolution,t_min,t
_max,t_exptime,t_resolution,em_min,em_max,em_res_power,o_ucd,pol_states,pol_xel,facility
_name,instrument_name,obs_release_date,t_plan_exptime,category,priority,execution_status)
VALUES (
58809.42866,'Cas A / Tycho SNR','21710053','16200160002',4.4687083,59.9733583,12.0,
<(4.4687083d , 59.9733583d),12d>,
720.0,58833.67603,58833.71689,3530.0,6.1E-5,1.241528257871965E-13,8.27685505247976
6E-11,0.00386,'photon.flux',NULL,0,'INTEGRAL','IBIS,
SPI',NULL,1500000.0,'Normal',0,'Scheduled');
```

- Please check pgsphere notation:
  <(**4.4687083d** , **59.9733583d**),**12d**>
  equivalent to CIRCLE('ICRS', ra, dec, radius)
- Note: If "d" is not added, values in radians.
- In case of doubt understanding a field, please also check ObsCore
  http://www.ivoa.net/documents/ObsCore/

Testing the system

- By using, e.g. TopCat, you can test your system by adding your service URL into the application.Generally, something like

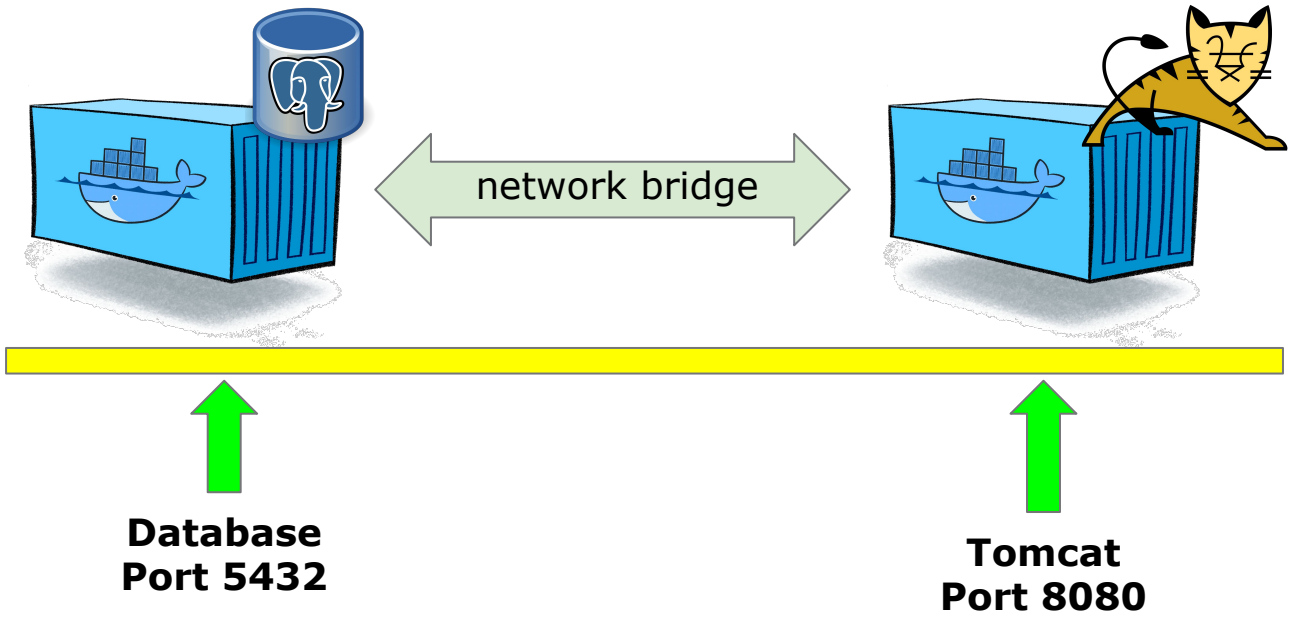
  http://localhost:8080/<tap_name>/tap

Try some queries, like:

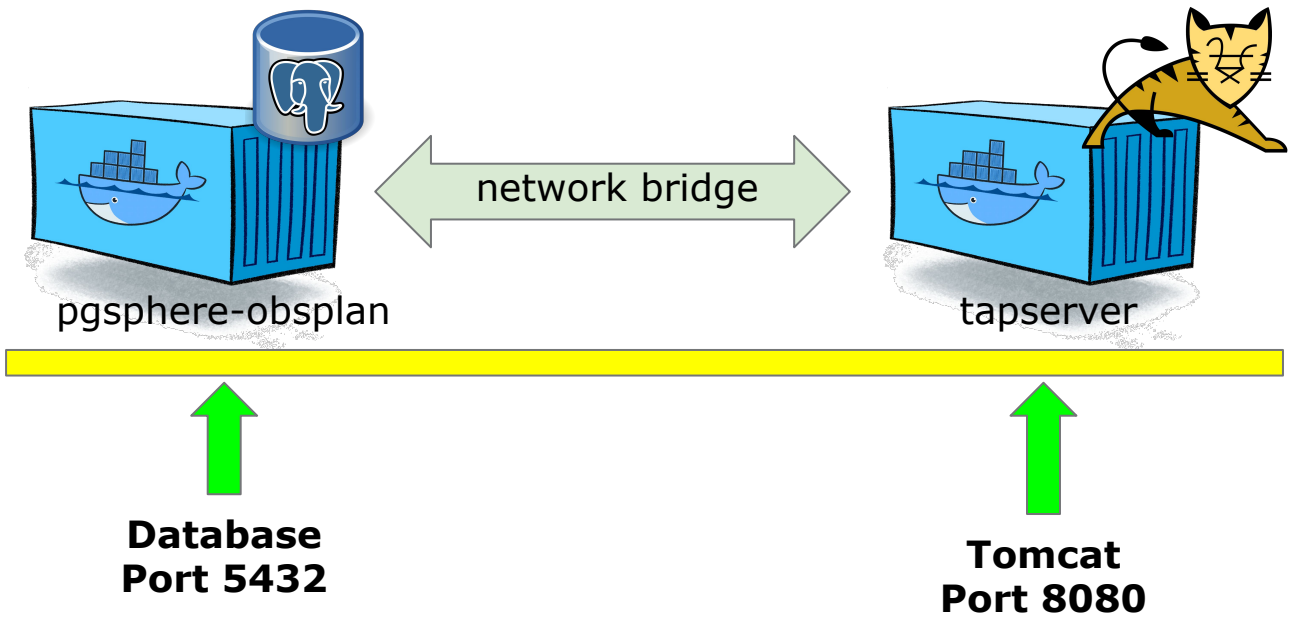
```
SELECT * FROM ivoa.obsplan
WHERE
t_max < 58502 AND
1=INTERSECTS(s_region,
CIRCLE('ICRS', 114.8251, 1.6179,
0.016666))
```

ObsLocTAP Docker short-cut

- Either to have a service to play with or to have a service without major effort, a couple of Docker instances are also provided:
  - PostgreSQL database with pgsphere installed, TAP_SCHEMA and ivoa.obsplan table with some rows
  - Tomcat server with taptuto installed
- As Docker containers are executed on a sandbox (isolated from the host machine) a network bridge is created to communicate them



ObsLocTAP by Docker start-up

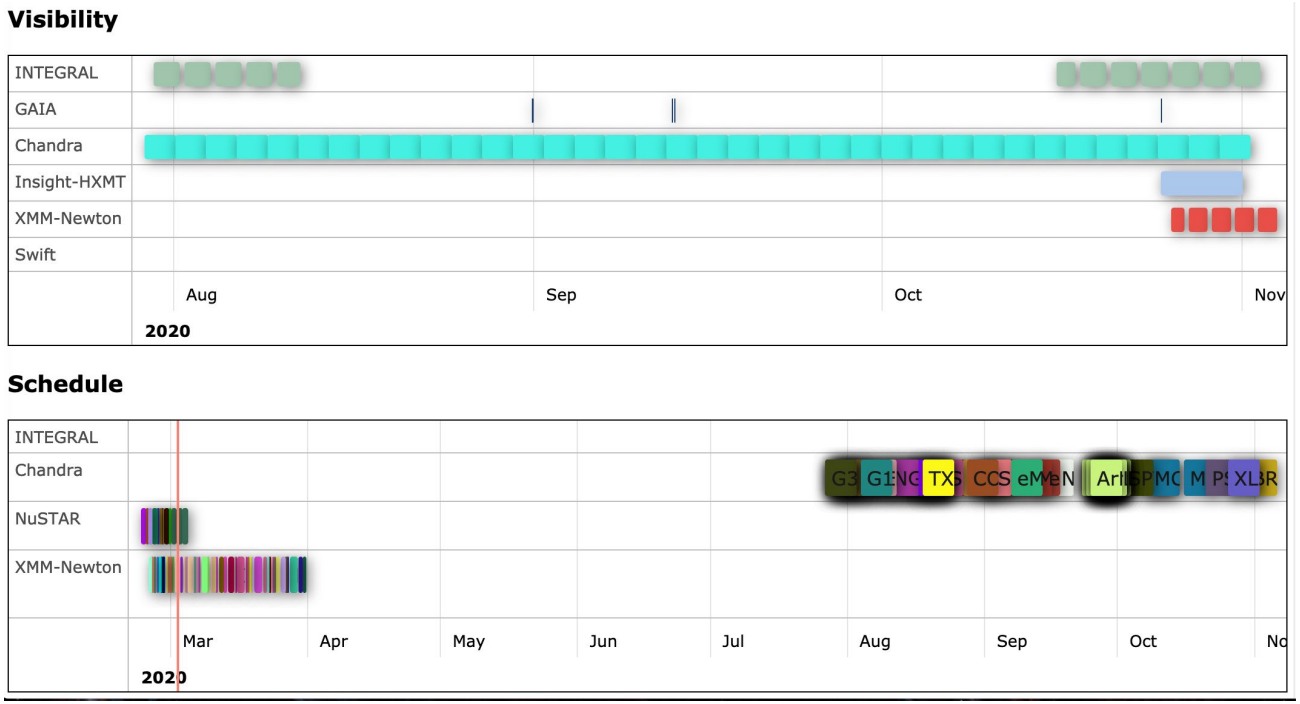



```
docker pull jsalgadodocker/pgsphere-obsplan:latest
docker pull jsalgadodocker/tapserver:latest
docker network create --driver=bridge db-network
docker image ls
docker run -p 8080:8080 --net=db-network --name tap jsalgadodocker/tapserver:latest
docker run -p 5432:5432 --net=db-network --name db jsalgadodocker/pgsphere-obsplan:latest
```

Reference implementation client

After successful implementation, contact E. Salazar for integration into:

http://integral.esa.int/toby/



Other clients expected: MySpaceCal (calendar for many missions), ESASky, etc