A short guide for publishing planetary data in the VO

Pierre Le Sidaner – Cyril Chauvin – B. Cecconi – S. Erard: Observatoire de Paris - Paris Astronomical Data Centre

1/4/19

Topic many time presented in Interop

Why repeated it one more time :

Protocol become more and more complex :

- TAP + ADQL + Sync & Async + Version 1.1.
- Datalink
- VOEvent (VTP)

If you want to develop, maintain and follow protocol evolution …

- Call the super team

Ask the people who know

Good client and framework :

I wont talk about client see Aladin, Topcat, Cassis, Amda ….

For publishing with TAP I have try :

- Gavo : DaCHS
- G. Mantelet libraries

- I wont talk about the others only because I didn't test them.

- Both are developed by very active and efficient person
- Large community use them, so there is knowledge base

- DaCHS is a framework (python)

- There is many facilities to extract metadata and publish a service

Initially deal with Postgres

First approach is to build a metadata database and publish it :

Now there is tutorial that make the job for you without knowing SQL

When you use Unstandardised data : Then write python reader or extract all in CSV format.

Super Markus write mixin for you to ingest data and he make the service working.

For free you have

- a publishing registry
- Simple web interface for presenting querying the service
- A huge amount of example

What's my added value in that

I am a poor programmer

I am not too smart

So we have developed How to for dummy

How to install in blind way

- Using Debian
- Using Docker

New service demo planets

From the Europlanet implementation session Done with C Brandt

DaCHS server :

-DaCHS* is a VO data publishing suite implementing all IVOA standard data services — particularly of our interest: EPN-TAP —, and also offers a rich data management interface.

Docker containers :

-Linux containers is a virtualisation technology to encapsulate an application in its own environment, isolated from the host Operating System; Containers are usually described as lightweight virtual machines. Docker containers is a particular implementation of Linux containers by Docker Inc.

* The GAVO Data Centre Help Suite (DaCHS) is developed by Markus Demleitner from the German Astronomical Virtual Observatory, http://adsabs.harvard.edu/abs/2014A&C.....7...27D

JACOBS
UNIVERSITY

- A Docker image of DaCHS is available at
  - chbrandt/dachs:latest

- “Dachs-on-Docker” is convenient as a sandbox, for testing Dachs and the deployment of new services and datasets, and especially if portability is of interest: Dachs on different Linux distributions, or MacOS, or Windows.

- Our goal here is to see how to ingest data on DaCHS and publish in the VO-Registry
  - We will define our site metadata and feed a database from a CSV file
  - We will discuss VO site concepts and register site service in the RofR
    - We will start by looking at DaCHS configuration files

- Keep in mind that Docker containers will start fresh every time one is initialised
  - Data files have to be shared from the host through “volumes”
  - Data may be persisted through “volume containers”

- Containers are meant to run one application at a time
  - We will use one container for DaCHS and one for Awstats

Docker Installation

To run a Docker container we need to install the Docker application/engine.
The official site provides installer for all popular operating systems*,**:

- https://docs.docker.com/install/

-If Docker is not yet installed, we will do it now.

-Then we download this tutorial data (docker_kit.zip) from:

- https://voparis-confluence.obspm.fr/download/attachments/42598508/docker_kit.zip

Content of docker_kit:

- ./img : logos
- ./logs : placeholder for DaCHS log files (will be used by Awstats)
- ./gavoetc : DaCHS configuration files
- ./mydisk : example of service (resource descriptor and data)
- ./conf : Awstats and Apache configuration files

* If you are using a Linux distribution, don't use the one provided by your distro repository;
** On Windows, we have tested and succeeded with Windows 10 only.

Config Files
(# comments, should not go)

Gavo.rc :

```
[general]
rootDir: /var/gavo
maintainerAddress: pierre.lesidaner@obspm.fr

[web]
bindAddress:
serverPort: 80
serverURL: http://127.0.0.1:80
sitename: mysitename
corsOriginPat: http.*                    # to allow cross site for javascript
logFormat: combined                      # to allow ip address in logs for awstats

[ivoa]
authority: mysite.institution            # this is your naming authority ID use in
registry
```

Config Files

Example of Defaultmeta.txt :

```
publisher: Paris Astronomical Data Centre
publisherID: ivo://vopdc.obspm

contact.name: PADC support team
contact.address: 77 av. Denfert Rochereau, 75014 Paris, FRANCE
contact.email: vo.paris@obspm.fr
contact.telephone: 0033140512082
creator.name: PADC
creator.logo: http://voparis-srv-paris.obspm.fr/logos/PADC_small.png

authority.creationDate: 2012-01-09T12:53:33
authority.title: Paris Astronomical Data Centre Authority
authority.shortName: PADCC
authority.description: The Paris Astronomical Data Centre project aims at providing VO
  access to its databases ressources, at participating to international standards
developments,
  at implementing VO compliant simulation codes, data visualization and analysis software.
  This Naming Authority is used to identify the resources provided by
  Paris Astronomical Data Centre at Observatoire de Paris, Paris, FRANCE
authority.referenceURL: http://padc.obspm.fr
authority.managingOrg: ivo://vopdc

organization.title: Observatoire de Paris
organization.description: Founded in 1667, the Observatoire de Paris is the largest national
  research center for astronomy.organization.referenceURL: http://www.obspm.fr

site.description: The Paris Astronomical Data Centre project aims at providing
  VO access to its databases ressources, at participating to international standards
developments,
  at implementing VO compliant simulation codes, data visualization and analysis softwares.
```

Run Dachs-on-Docker

Let's start the docker container :

*For simplicity, let's call "dk_path" the path where for `docker_kit` content is;
For example,
my_path=/user/home/docker_kit

```
$ cd my_path
$ docker run -it -p 80:80 --name mydachs \
      --volume my_path/mydisk/planets:/var/gavo/inputs/planets \
            --volume my_path/logs/:/var/gavo/logs \
      chbrandt/dachs:latest
```

Comments :

'-p' : redirection of port from the host to the container.
In this case, we are saying that any request directed to our host' port 80 should be redirected to the container' port 80. If your host is already using port 80 (with another webserver, for example) you opt for another port (e.g., 88) by saying '-p 88:80'.

'--volume' : defines a directory to be shared between host and container

'-- name' : give a name to your container, it helps you on managing it

'chbrandt/dachs:latest' is effectively the image of the DaCHS container on DockerHub

New service demo planets

Service definition :

In mydisk/planets :

- ‘q.rd’ is the Resource Descriptor, where the service as a whole is defined (data, metadata, interface).

- ‘data/Masses2.csv’ provides our service data, to be ingested in our database

- ‘planets’, the name of the directory, has a tight relation to the database schema.

  Usually we will match the name of the service directory (here ‘planets’) with the name of the database schema (the first element of the Resource Descriptor, ‘q.rd’).
  That been said: don’t use whitespaces, periods (‘.’) or capital letters in it.

New service demo planets

Service definition :

The ‘q.rd’ is a very structured (XML) file with multiple blocks defining the database, how to import the data into it and the metadata associated to the service. It is composed by three major blocks:

```
<resource schema=…>

    <!- - 'meta' fields describe the service metadata, e.g., 'title' -->
    <meta name=…>…</meta>
    …

    <!- - 'table' defines the database table, its columns and content therein -->
    <table id="epn_core" onDisk="true" adql=“True”>
        …
    </table>

    <!- - 'data' defines how to ingest data into the database -->
    <data id="import">
        …
    </data>

</resource>
```

New service demo planets (2)

Service is in mydisk/planets :

To assign a constant value to a column:

```
<make table="epn_core">
    <rowmaker idmaps="*">
        <var key="dataproduct_type">"ci"</var>
```

To associate a name of a column in csv file @granule_gid to a filed in epn-tap

```
    <apply procDef="//epntap2#populate-2_0" name="fillepn">
        <bind name="granule_gid">@granule_gid</bind>
```

To publish the specific table epn_core:

```
<data id="collection" auto="false">
    <make table="epn_core"/>
    <publish/>
```

Docker Installation (3)

Once we have the container “mydachs” running, we can push DaCHS configuration files from our host into our container with the following commands*

```
sudo docker exec -it mydachs mkdir /var/gavo/web/nv_static/img
sudo docker exec -it mydachs chown dachsroot:gavo /var/gavo/web/nv_static/img
sudo docker cp my_path/gavoetc/gavo.rc mydachs:/etc/gavo.rc
sudo docker cp my_path/gavoetc/defaultmeta.txt mydachs:/var/gavo/etc/defaultmeta.txt
sudo docker cp my_path/img/logo_big.png mydachs:/var/gavo/web/nv_static/img/logo_big.png
sudo docker cp my_path/img/logo_medium.png \
                mydachs:/var/gavo/web/nv_static/img/logo_medium.png
```

* Alternative to this process would be to mount the entire ‘my_path’ to the container and move all files from inside the container. We have chosen to use this verbose form to highlight the flexibility we have to change the state of the container without effectively going inside it.

Docker Installation (4)

Import data and run the DaCHS/VO service

```
sudo docker exec -it mydachs /etc/init.d/postgresql restart
sudo docker exec -it --user dachsroot mydachs  gavo imp /var/gavo/inputs/planets/q.rd
sudo docker exec -it --user dachsroot mydachs  gavo pub //services
sudo docker exec -it --user dachsroot mydachs  gavo pub //tap
sudo docker exec -it --user dachsroot mydachs  gavo pub /var/gavo/inputs/planets/q.rd
sudo docker exec -it mydachs gavo serve restart
```

Comments :

-gavo imp q
- Allow to run the metadata ingestion

-gavo pub //services
- read defaultmeta.txt and build the publishing registry with naming auth.

-gavo pub //tap
- publish the tap service in the local registry

-gavo pub q.rd
- publish the specific table of the service as a collection of data

You can test with your browser http://localhost
then go to ADQL query and type : select * from planets.epn_core then press Go

Register the service

The Virtual Observatory have a “yellow pages” mechanism called registry

-You can register your service using a web interface
-Or you can use DaCHS itself to publish as the system and configuration of local registry is done.
    -To test http://127.0.0.1/oai.xml

    Don't worry about the error normal url syntax should follow oai syntax
    Click on All “identifiers defined here” you will see your service registered

As it is a test and we don’t have a public IP we can not register the service.
But when it will be available on a public server go to registry of registry (RofR)
    - http://rofr.ivoa.net/
to make your publishing registry harvested.
You give the url of your DaCHS server with ‘/oai.xml’ at the end.

Docker installation of Awstats

Build the container from debian image:

```
sudo docker run -it --name myawstats -p 8080:80 -v my_path/logs/:/var/gavo/logs/ debian
sudo docker exec -it myawstats apt-get -y update
sudo docker exec -it myawstats apt-get -y upgrade
sudo docker exec -it myawstats apt-get -y install awstats
sudo docker exec -it myawstats apt-get -y install apache2
sudo docker exec -it myawstats apt-get -y install geoip-database
sudo docker exec -it myawstats apt-get -y install libgeo-ipfree-perl
sudo docker cp mypath/conf/awstats.dachs.conf myawstats:/etc/awstats/awstats.dachs.conf
sudo docker cp mypath/conf/25-myserver.mydomain.conf \
      myawstats:/etc/apache2/sites-enabled/25-myserver.mydomain.conf
```

-We built a container (from ‘debian’) binding container's port 80 to host’s 8080
- -We install awstats and apache2
- -Copy Awstats config files, that you have to edit first:
  - "myserver" to your server name and "mydomain" to your domain
- -Copy Apache config files, that you have to edit first:
  - change "voparis-tap-planeto.obspm.fr" to your server name

Run Awstats and param cron

We have to initiate the stats and make :

```
sudo docker cp mypath/conf/run_awstats myawstats:/etc/cron.daily/run_awstats
sudo docker  exec -it myawstats chown root:root /etc/cron.daily/run_awstats
sudo docker  exec -it myawstats chmod 777 /etc/cron.daily/run_awstats
sudo docker  exec -it myawstats /etc/cron.daily/run_awstats
sudo docker  exec -it myawstats a2enmod cgi
sudo docker  exec -it myawstats /etc/init.d/apache2 restart
```

Docker survival kit

Minimum command :

sudo docker ps -a	list docker container
sudo docker container rm docker_ID	remove container (stop to stop)
docker start -a mydachs	to start container
sudo docker exec -it mydachs bash	To open a shell on a running container