PUBLISHING COMBINED IMAGE

AND

SPECTRAL DATA PACKAGES

1 Introduction

ESO has developed tools for publishing images and spectra compliant with SIAP and SSAP Virtual Observatory standards. In this exercise data from the Great Observatories Origin Deep Survey (GOODS) are used: Near infrared images (ISAAC) as wells as multi-object, optical to NIR spectra (FORS2) in the Chandra Deep Field South (CDFS)

Workflow overview:

- Software Installation
- Metadata Extraction and Ingestion Process (Image and Spectrum data files)
  - Database setup
    - create SQL tables
  - Mapping editor
    - get sample FITS file header per data type
    - define mappings between FITS keywords and IVOA models metadata
    - test mappings against sample FITS file headers
  - MEx
    - ingest FITS into database
- Access the data

  Query SIA and SSA services at ESO

We prepared two exercises, one for images and one fore spectra to be able to select the type of data we want to use and it only affects the metadata extraction and ingestion process. This was done for simplicity, nevertheless all different types of data can be ingested at the same time.

To publish data in the VO the first step is to be familiar with VO standards, we suggest you to take a look at the standards and documentation at the IVOA (**I**nternational **V**irtual **O**bservatory **A**lliance) site http://www.ivoa.net/Documents.

Through the exercise the following VO protocols will be used:

SIAP

This protocol defines an uniform interface for retrieving image data from a variety of astronomical image repositories. The interface is meant to be reasonably simple to implement by service providers. A query defining a rectangular region on the CAR frame is used to query for candidate images. The service returns a list of candidate images formatted as a VOTable. For each candidate image an access reference URL may be used to retrieve the image. Images may be returned in a variety of formats including fits and various graphics formats.

Working Draft: http://www.ivoa.net/Documents/latest/SIA.html

SSAP

This protocol defines an uniform interface to remotely discover and access one dimensional spectra. SSA is a member of an integrated family of data access interfaces altogether comprising the Data Access Layer (DAL) of the IVOA. SSA is based on a more general data model capable of describing most tabular spectrophotometric data, including time series and spectral energy distributions (SEDs) as well as 1-D spectra.

Recommendation: http://www.ivoa.net/Documents/latest/SSA.html

2 Installation Notes

The instructions in this hands-on are targeted to a Linux system running a bash shell. If you use another shell, you will have to adapt some commands (notably the setting of environment variables).

Not all steps can be run under Windows. Each step indicates on which systems it works on.

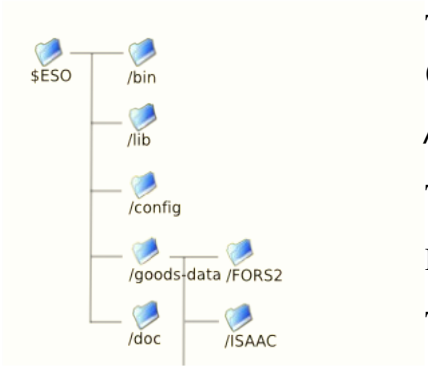
All the material needed for this exercise can be downloaded from:

http://cds.u-strasbg.fr/twikiDCA/bin/view/EuroVODCA/ESOWorkflowVODaySofia

Follow the software installation instructions for common packages for the workshop and copy the tar or zip file in your working directory:

```
tar -xvf workshop_eso.tar
export ESO="{YOUR_WORKING_DIRECTORY}"
chmod -R +w $ESO
```

You will get an structure similar to the one below:



The ***/bin*** directory contains the programs used for this exercise (except MySQL, JAVA and convert).

/lib owns the libraries needed by the programs.

The configuration files are in ***/config***.

Data are stored in ***/data*** under different directories.

This manual can be found at ***/doc***.

2.1 Common packages

The required program and libraries are either distributed in the workshop common distribution (MySQL 5, Tomcat 5.0) or in our specific distribution, either as binary or source (see next section for details).

The common programs should have been properly installed and configured, including setting the following environment variables:

	environment variable
Tomcat 5.0.28	$CATALINA_HOME
MySQL 5.0.41	$MYSQL

E.g. (under Linux and bash):

```
export CATALINA_HOME="$HOME/common/jakarta-tomcat-5.0.28"
export MYSQL="$HOME/common/mysql"
```

E.g. (under Windows):

```
set CATALINA_HOME="C:\Program Files\Apache Software Foundation\Tomcat 5.0"
set MYSQL="C:\Program Files\MySQL\MySQL Server 5.0"
```

We expect that the MySQL root user has no password. If that is not the case, please reset it:

```
$MYSQL/bin/mysqladmin -u root -p password ''
```

Under MacOSX, MySQL needs to be located under /usr/local/mysql; if not:

```
sudo ln -s $MYSQL /usr/local/mysql
```

Also under MacOSX, the following fix must be applied:

```
sudo ln -s /usr/local/mysql/lib /usr/local/mysql/lib/mysql
```

Set some environment variables to be used during the exercise are set by (look at the file contents and edit if needed):

```
source $ESO/setup/setup.env
```

On MacOSX, some environment variables must be different, so load also:

```
source $ESO/setup/setup.env.mac
```

Note that if you compile some packages, you should start a new session and set all these environment variables when finished.

Once you finish the setup defined on these installation notes, you should start a new session and set all these environment variables and source the setup.env file.
To streamline it, you should edit the setup.env file to set the JAVA_HOME, CATALINA_HOME, MYSQL and ESO variables at the top.

2.2 Specific packages

Java 1.4 and Python 2.4 are required by MEx. MEx is not supported on Windows, so on Windows you will not be installing those two packages.

2.2.1 Java 1.4

```
cd $ESO/opt
$ESO/setup/dependencies/binaries/Java/j2sdk-1_4_2_14-linux-i586.bin
```

(Note: under MacOSX you should already have a Java 1.4 installation)

2.2.2 Python 2.4

Install Python 2.4 with dependencies. Unpack one of the pre-compiled bundles:

```
cd $ESO
tar xfz setup/dependencies/binaries/Python/python-eso-{YOUR_PLATFORM}.tar.gz
```

The previous bundles have not been extensively tested on all platforms; should you run into troubles, you can compile them yourself from source:

```
cd $ESO/setup
./dependencies/setup.sh | tee dependencies/setup.out
```

On Windows we won't use Python for the ingestion step, so no dependencies are required. Install Python from $ESO/setup/dependencies/binaries/Python/python-2.4.4.msi

3 Database setup

Create the database (called exercise_eso)

```
$MYSQL/bin/mysql –u root
   CREATE DATABASE exercise_eso;
   USE exercise_eso
   GRANT ALL ON exercise_eso.* TO ''@'localhost';
   exit

cd $ESO/config
$MYSQL/bin/mysql --password="" –D exercise_eso
   source exercise_eso_crebas.sql
   exit
```

If mysql complains about not finding the user when assigning permissions, create it by issuing the command:

```
CREATE USER ''@'localhost';
```

4 Image Exercise

4.1 Ingestion (for Windows)

The tools to ingest the data do not work on Windows. The only thing that can be done is to move the data files in an appropriate location and populate the database, to be able to query and access the files.

To move the data, execute
```
%ESO%\bin\shuffleData.bat
```

To populate the database:
```
%MYSQL%\bin\mysql –u root < %ESO%\extra\database\allfiles.sql
```

4.2 Ingestion (Except for Windows)

We will only ingest images (FITS files), but most procedures apply to all file types.

4.2.1 Inventory: catalogue the files

10 min

Once we have all the files that we want to publish, the next step is to catalogue them according to their type and internal FITS structure. The Catalogue Builder tool aids in this, first by creating an empty directory structure, and then by creating a catalogue file that categorises all files according to their location in the directory structure.

To create the empty structure and place files to appropriate locations, run the script:

```
$ESO/bin/shuffleData-images.sh (under Windows %ESO%\bin\shuffleData.bat)
```

This will create a data folder containing symlinks to all image files. This folder will be used to serve the data by the SIA service. It will also create data-images, used for the ingestion step. Note that there is need for these two separate folders when several type of data are to be ingested at once, although it is not the case here we keep them to have a consistent workflow.

To generate the catalogue, run the catalogue builder on build mode:

```
cd $ESO
mkdir filelists
java -jar bin/catalogueBuilder.jar catbuild -file filelists/filelist-images.xml
-dir data-images
```

An HTML version of the catalogue is also generated, so that you can check that all files were correctly categorised.

The catalogue file generated in this step can be found (among the ones from other exercises) under $ESO/extra/filelists. If you had problems running this step:

```
mkdir -p $ESO/filelists/
cp $ESO/extra/filelists/filelist-images.xml $ESO/filelists/
```

4.2.2 Header extraction

05 min

For each type of file (and with a different FITS header structure), a mapping from FITS headers to database model must be defined. To aid this operation, we should extract a FIST header from each different type of file. This sample header will be used to preview the mappings result:

```
mkdir $ESO/headers
dfits -x 0 $ESO/goods-data/ISAAC/GOODS_ISAAC_03_H_V1.5.fits >
$ESO/headers/isaac.txt
```

The header file generated in this step can be found (among the ones from other exercises) under $ESO/extra/headers. If you had problems running this step:

```
mkdir -p $ESO/headers/
cp $ESO/extra/headers/isaac.txt $ESO/headers/
```

4.2.3 Datamodel mapping

20 min

Create the folder to store the mappings

```
mkdir $ESO/mappings
```

We use the mapping editor to obtain the mapping for our files (to identify the fits keywords with the columns in the database).

First start Tomcat

```
$CATALINA_HOME/bin/startup.sh
```

Open a browser and go to http://localhost:58080/mapedit/

- Select the type of data to configure (image.stacked in our case)
- Browse the header file you just created and click “Save & Validate”
- Edit the expressions (mappings between FITS keywords and datamodel items) until they are correct. Rules can be simple one-to-one mappings from FITS keywords, contain arithmetic, constants, unit conversions[1] and more.
- Validate your rules, using the sample FITS header. This way you have a preview to better ascertain the correctness of your mappings.
- When finished editing, either copy the mapping rules (text area on the bottom of the page) or send it by e-mail to yourself. Store it under $ESO/mappings/isaac.txt. This mapping will be used by MEx during the ingestion process.

The mapping file generated in this step can be found (among the ones from other exercises) under $ESO/extra/mappings. If you had problems running this step:

```
mkdir -p $ESO/mappings
cp $ESO/extra/mappings/isaac.txt $ESO/mappings/
```

You can also try the ESO stand-alone mapping editor available at:
http://vo.eso.org:8080/mapedit

You can check the list of the minimum set of information ('data items') requested at ESO for the data to be published, images in our case:

http://archive.eso.org/cms/eso-data/data-submission/image-guide

4.2.4 Homogenization

10 min

Ingest the package into the database. MEx, the Metadata Extractor, runs on the data files given the catalogue file and the mappings.

```
cd $ESO
./mex/persist_exercise_eso.py --itemdef-file=config/modelitem_definitions.txt --
verbosity=0 --free-units --host localhost --user root --database exercise_eso -l
filelists/filelist-images.xml -m mappings/isaac.txt
```

1 using the Unit Conversion Java library from CDS (http://cdsweb.u-strasbg.fr/cdsdevcorner/units.gml)

If you are using a filelist that you found under the /extra folder, you'll have to add an argument to MEx specifying where the files are located:

```
-d $ESO/data
```

The data inserted into the database in this step can be found (among the ones from other exercises) under $ESO/extra/database. If you had problems running this step:

```
$MYSQL/bin/mysql -u root < $ESO/extra/database/images.sql
```

5 Spectral Exercise

5.1 Ingestion (for Windows)

The tools to ingest the data do not work on Windows. The only thing that can be done is to move the data files in an appropriate location and populate the database, to be able to query and access the files.

To move the data, execute

```
%ESO%\bin\shuffleData.bat
```

To populate the database:

```
%MYSQL%\bin\mysql -u root < %ESO%\extra\database\allfiles.sql
```

5.2 Ingestion (Except for Windows)

In this simple exercise we will only ingest spectra (FITS files), but most procedures apply to all file types.

5.2.1 Inventory: catalogue the files

10 min

Once we have all the files that we want to publish, the next step is to catalogue them according to their type and internal FITS structure. The Catalogue Builder tool aids in this, first by creating an empty directory structure, and then by creating a catalogue file that categorises all files according to their location in the directory structure.

To create the empty structure and place files to appropriate locations, run the script:

```
$ESO/bin/shuffleData-spectra.sh  (under Windows %ESO%\bin\shuffleData.bat)
```

This will create a data folder containing symlinks to all spectra files. This folder will be used to serve the data by the SSA sevice. It will also create data-spectra, used for the ingestion step. Note that there is a need for these two separate folders when several type of data are to be ingested at once, although it is not the case here we keep them to have a consistent workflow.

To generate the catalogue, run the catalogue builder on build mode:

```
cd $ESO
mkdir filelists
java -jar bin/catalogueBuilder.jar catbuild -file filelists/filelist-spectra.xml
-dir data-spectra
```

An HTML version of the catalogue is also generated, so that you see that all files were correctly categorised.

The catalogue file generated in this step can be found (among the ones from other exercises) under $ESO/extra/filelists. If you had problems running this step:

```
mkdir -p $ESO/filelists/
cp $ESO/extra/filelists/filelist-spectra.xml $ESO/filelists/
```

5.2.2 Header extraction

05 min

For each type of file (and with a different FITS header structure), a mapping from FITS headers to database model must be defined. To aid this operation, we should extract a fits header from each different type of file. This sample header will be used to preview the mappings result:

```
mkdir $ESO/headers
dfits -x 0 $ESO/goods-data/FORS2/GOODS_FORS2_GDS_J033202.99-
274301.2_904509_V2.0.fits > $ESO/headers/fors2.txt
```

The header file generated in this step (including the ones from other exercises) can be found under $ESO/extra/headers. If you had problems running this step:

```
mkdir -p $ESO/headers/
cp $ESO/extra/headers/fors2.txt $ESO/headers/
```

5.2.3 Datamodel mapping

20 min

Create the folder to store the mappings

```
mkdir $ESO/mappings
```

We use the mapping editor to obtain the mapping for our files (to identify the fits keywords with the columns in the database).

First start Tomcat

```
$CATALINA_HOME/bin/startup.sh
```

Open a browser and go to http://localhost:58080/mapedit/

- Select the type of data to configure (spectra.1d)
- Browse the header file you just created and click "Save & Validate"
- Edit the expressions (mappings between FITS keywords and datamodel items) until they are correct. Rules can be simple one-to-one mappings from FITS keywords, contain arithmetic, constants, unit conversions[2] and more.
- Validate your rules, using the sample FITS header. This way you have a preview to better ascertain the correctness of your mappings.
- When finished editing, either copy the mapping rules (text area on the bottom of the page) or send it by e-mail to yourself. Store it under $ESO/mappings/fors2.txt. This mapping will be used by MEx during the ingestion process.

The mapping file generated in this step can be found (among the ones from other exercises) under $ESO/extra/mappings. If you had problems running this step:

```
mkdir -p $ESO/mappings
cp $ESO/extra/mappings/fors2.txt $ESO/mappings/
```

You can also try the ESO stand-alone mapping editor available at:
http://vo.eso.org:8080/mapedit

You can check the list of the minimum set of information ('data items') requested at ESO for the data to be published, spectra in this case:

http://archive.eso.org/cms/eso-data/data-submission/spectrum-guide

You can also find those lists under $ESO/extra/submission/

2 using the Unit Conversion Java library from CDS (http://cdsweb.u-strasbg.fr/cdsdevcorner/units.gml)

5.2.4 Homogenization

10 min

Ingest the package into the database. MEx, the Metadata Extractor, runs on the data files given the catalogue file and the mappings.

```
cd $ESO
./mex/persist_exercise_eso.py --itemdef-file=config/modelitem_definitions.txt --
verbosity=0 --free-units --host localhost --user root --database exercise_eso -l
filelists/filelist-spectra.xml -m mappings/fors2.txt
```

If you are using a filelist that you found under the /extra folder, you will have to add an argument to MEx specifying where the files are located:

```
-d $ESO/data
```

The data inserted into the database in this step can be found (among the ones from other exercises) under $ESO/extra/database. If you had problems running this step:

```
$MYSQL/bin/mysql -u root < $ESO/extra/database/spectra.sql
```

6 Access the Data

The following step in the workflow would be to create the services to access the data we just ingested in the database. We need to create a service that makes a query to our database following the VO standards, SIAP and SSAP.

We do not have enough time in this hands-on session for the creation of the services. We will use the services available at ESO to query exactly the same data we just ingested, stored in a similar database (mysql table).

6.1 ESO SIA Service

The SIA service created at ESO to query this data can be tested using the following URLs:

Query the service metadata:

http://vops2.hq.eso.org/cgi-bin/eso.footprint?VERSION=1.0&FORMAT=METADATA

The metadata query is mainly intended for use by a central registry that will collect this information so that users

can search for services by their capabilities. It is also useful for communicating non-standard input parameters. The response VOTable obtained by a metadata request is the same as normal query response except that it contains no table rows. All input and output parameters to be available to clients of the service must be listed, including required parameters, optional parameters, and non-standard parameters specific to the service.

Try a simple query:

http://vops2.hq.eso.org/cgi-bin/eso.footprint?VERSION=1.0&POS=53,-27&SIZE=5

Some examples of VOTables obtained trying different queries can be found under $ESO/extra/sia

6.2 ESO SSA Service

The SIA service created at ESO to query this data can be tested using the following URLs:

Query the service metadata:

http://archive.eso.org/apps/ssaserver/ForsProxySsap?REQUEST=queryData&FORMAT=METADATA

Try a query with all possible query parameters:

You can make a positional query:

http://archive.eso.org/apps/ssaserver/ForsProxySsap?REQUEST=queryData&POS=53,-27&SIZE=0.7&BAND=&TIME=&FORMAT=all

Or try a query by time, giving a range-list:

http://archive.eso.org/apps/ssaserver/ForsProxySsap?REQUEST=queryData&TIME=2004-01-01/2006

Queries on any combinations of the parameters can be done.

Some examples of VOTables obtained trying different queries can be found under $ESO/extra/ssa

7 Optional: try with your data

You can reproduce this exercise with your own data.

If you were not able to install all the components needed, you can try just to map your own data using the ESO stand-alone mapping editor available at:

http://vo.eso.org:8080/mapedit

The mapping editor helps defining the FITS keyword mapping by running the mapping engine (MEx) against a sample FITS header.

Select the type of data for the mapping. Upload or copy the FITS header and validate it. All required metadata must have a value to have your dataset properly described. The values are obtained by executing the rule expression. A default rule expression is predefined for all items. After validation remove the invalid rules. The validation runs all rules against the sample FITS header, and the resulting values are displayed on the test result column. The mapping rules used are shown at the end of the page, in a text box that is updated whenever mappings are validated.