Although ECOMS-UDG provides a web interface to explore and access the datasets, it is strongly recommented the use of OPeNDAP (a.k.a. DODS) client libraries to remotely access the data from scientific computing environments (R, Matlab, Python, etc.). In order to facilitate this task a number of utilities for (remote) data access have been developed as part of the ECOMS-UDG.

The ecomsUDG.Raccess **R package** is the main of these tools, and it is based on the ?NetCDF Java OPeNDAP client, using the rJava R package (a similar approach is followed in the **Matlab** interface). Alternatively, the most recent NetCDF library versions provide access to OPeNDAP datasets (this is the solution for the **Python** implementation).

IMPORTANT NOTE: Unlike the R package, which will be continuously updated as part of the data management activities, the Python and Matlab functions are not guaranteed to be maintained and updated with the same regularity, and therefore the use of the R environment is encouraged in the framework of ECOMS. In particular, the different datasets are not homogenized in the Matlab and Python versions and the only variables available are those originally available in the corresponding datasets (see the list of available variables? for details).

- Python Interface
- MatLab Interface

Low-leves interface to OpenDAP service

Although the TDS provides a web interface to explore and access the datasets (shown in web access section), it is strongly recommented the use of OPeNDAP (a.k.a. DODS) client libraries to remotely access the data from scientific computing environments (R, Matlab, Python, etc.). For instance, the R function provided in this tutorial is based on the *NetCDF Java* OPeNDAP client[1], using the rJava R package (a similar approach is been also made for the Matlab implementation). Alternatively, the most recent *NetCDF library* versions provide access to OPeNDAP datasets (this is the solution for the Python implementation). In the following, we show a simple example of data access using the R package developed as part of the data portal. In particular the *System4* datasets can by directly accessed using the loadSystem4 function, allowing the retrieval of slices for a particular variable in any of the dataset dimensions (member/space/runtime/time). Note that a more ellaborated worked example using R is shown in the R example section. Moreover, for a better understanding of the datasets structure, the use of the web interface for the OPeNDAP service is also illustrated web access section.

Accessing to the Data portal using Python (Pydap version)

This section **still** needs revision

```
[user@host ~]$ pip install Pydap
.....................................................................
[user@host ~]$ python
Python 2.7.2 (default, Mar  3 2012, 10:45:44)
[GCC 4.1.2 20080704 (Red Hat 4.1.2-48)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

The new registration process impose a more elaborated http requests. The PyDap package is able to follow redirects but cookies are ignored. The UDG python package has a been released to help users handle the new authentication service. Download the UDG Python package

```
>>> from udg import install_udg_client
>>> install_udg_client()
>>> from pydap.client import open_url
>>> udg_dataset='system4/System4_Seasonal_15Members.ncml'
>>> url= 'http://%s:%s@meteo.unican.es/tds5/dodsC/%s' % ('YOUR_USERNAME','YOUR_PASSWORD',udg_dataset)
>>> dataset = open_url(url)
>>> print type(dataset)
<class 'pydap.model.DatasetType'>
>>> print dataset.keys()
['lat', 'lon', 'run', 'time', 'time1', 'time2', 'member', 'Maximum_temperature_at_2_metres_since_last_24_hours_surface', '
>>> MN2T24 = dataset['Minimum_temperature_at_2_metres_since_last_24_hours_surface']
>>> print MN2T24.dimensions
('member', 'run', 'time', 'lat', 'lon')
>>> print MN2T24.shape
(15, 360, 215, 241, 480)
>>> arr = MN2T24[0,11:360:12,31:62,66,475]
>>> print numpy.squeeze(numpy.mean(arr,2))
```

```
[ 270.79171753  273.29437256  271.56661987  271.03707886  271.82745361
 272.49279785  271.48086548  268.59121704  271.53125     273.82156372
 270.99401855  274.23626709  270.99328613  271.56115723  273.98986816
 270.50756836  272.45046997  270.65560913  271.31182861  272.77200317
 273.4359436   271.85021973  273.39648438  274.16384888  269.98248291
 271.30166626  273.11950684  271.27301025  272.29147339  270.46688843]
```

Accessing to the Data portal using Octave

This section needs revision and integrtion with the Matlab example section

```
>> ver
---------------------------------------------------------------------
GNU Octave Version 3.6.1
GNU Octave License: GNU General Public License
Operating System: unknown
---------------------------------------------------------------------
```

```
>> urlwrite('http://www.meteo.unican.es/work/netcdfAll-4.3.jar','netcdfAll-4.3.jar')
>> javaaddpath('./netcdfAll-4.3.jar');
>> javaMethod('setGlobalCredentialsProvider','ucar.nc2.util.net.HTTPSession',javaObject('ucar.nc2.util.net.HTTPBasicProvid
>> ncfile = javaMethod('openDataset','ucar.nc2.dataset.NetcdfDataset','http://www.meteo.unican.es/tds5/dodsC/system4/Syste
>> v = ncfile.findVariable('Minimum_temperature_at_2_metres_since_last_24_hours_surface');
>> disp(v.getDimensions.toString)
[   member = 15;,    run = 360;,    time = 215;,    lat = 241;,    lon = 480;]
>> d = v.read('0,11:359:12,31:61,66,475');
>> tmp = javaObject('org.octave.Matrix',d.reduce.copyToNDJavaArray);
>> oldFlag = java_convert_matrix (1);
>> octaveMatrix = tmp.ident(tmp);
[ (30 by 31) array of double ]
>> disp(squeeze(mean(octaveMatrix,2))')
Columns 1 through 13:

  270.79   273.29   271.57   271.04   271.83   272.49   271.48   268.59   271.53   273.82   270.99   274.24   270.99

Columns 14 through 26:

  271.56   273.99   270.51   272.45   270.66   271.31   272.77   273.44   271.85   273.40   274.16   269.98   271.30

Columns 27 through 30:

  273.12   271.27   272.29   270.47
```

Accessing to the Data portal using Matlab

This section needs revision]

```
>> ver
-------------------------------------------------------------------------------------
MATLAB Version 7.8.0.347 (R2009a)
MATLAB License Number: 161051
Operating System: Microsoft Windows Vista Version 6.1 (Build 7601: Service Pack 1)
Java VM Version: Java 1.6.0_04-b12 with Sun Microsystems Inc. Java HotSpot(TM) 64-Bit Server VM mixed mode
-------------------------------------------------------------------------------------
>> javaaddpath('http://www.meteo.unican.es/work/netcdfAll-4.3.jar');
>> %javaaddpath('ftp://ftp.unidata.ucar.edu/pub/netcdf-java/v4.3/netcdfAll-4.3.jar');
```

```
>> import ucar.nc2.util.net.* %this will download the netcdfAll-4.3.jar
>> HTTPSession.setGlobalCredentialsProvider(HTTPBasicProvider('username','password'));
>> import ucar.nc2.*;
>> import ucar.nc2.dataset.*;
>> ncfile = NetcdfDataset.openDataset('http://www.meteo.unican.es/tds5/dodsC/system4/System4_Seasonal_15Members.ncml');
>> v = ncfile.findVariable('Minimum_temperature_at_2_metres_since_last_24_hours_surface');
>> disp(v.getDimensions)
[   member = 15;,    run = 360;,    time = 215;,    lat = 241;,    lon = 480;]
>> data = v.read('0,11:359:12,31:61,66,475').copyToNDJavaArray();
>> disp(squeeze(mean(data,3)))
 Columns 1 through 13

 270.7917  273.2944  271.5666  271.0371  271.8275  272.4928  271.4809  268.5912  271.5313  273.8216  270.9940  274.2363  2

 Columns 14 through 26

 271.5612  273.9899  270.5076  272.4505  270.6556  271.3118  272.7720  273.4359  271.8502  273.3965  274.1638  269.9825  2

 Columns 27 through 30

 273.1195  271.2730  272.2915  270.4669
```

1. 1. ?http://www.unidata.ucar.edu/software/netcdf-java/documentation.htm