WCS primer

In this primer we will demonstrate how to obtain a small set of data from a large datasets with the OGC Web Coverage Service (WCS) protocol. We will show that ordering data through WCS is just as easy as buying an ice cream. Bear in mind the construction of a WCS request depends on the version supported by the data provider. Currently the versions are 1.0.0, 1.1.1 and 2.0.1.

Find a WCS server

Find a data web source that hosts a WCS server (*go to an ice cream vendor*). We will use the Shuttle Radar Topography Mission datasets hosted at the Woods Hole Institute THREDDS OPeNDAP server as example in this primer. THREDDS OPeNDAP server can be configured to serve WCS for orthogonal datasets with proper geographic information. Here is a list with WCS servers. For the Netherlands search the PDOK WCS services for Nertherlanda, for instance AHN25.

Request an overview of the content of a WCS server

Ask for what the server has to offer (*see which flavours he has and which kind of cups*). You need to add the following mandatory <keyword,value> pairs to the base server url, separated by an &, e.g.: `?service=WCS&request=GetCapabilities`.

keyword	value	source
service	WCS	Mandatory WCS standard value
request	GetCapabilities	Mandatory WCS standard value

This procedure works for all subsequent <keyword,value> pairs in this primer:
`http://geoport.whoi.edu/thredds/wcs/bathy/srtm30plus_v6?service=WCS&request=GetCapabilities`

This url will return an xml file that contains an inventory of the available datasets. (You can also request available datasets for one WCS version only by appending the optional `version` keyword.)

Example for 2.0.1 request on RWSProjectArchive geoserver.

2.0.1 GetCapabilities

```
https://rwsprojectarchief.openearth.nl/geoserver/mepduinen/wcs?service=WCS&version=2.0.1&request=GetCapabilities
```

Inspect the overview of the content of a WCS server

Look at what versions of WCS the server has to offer (*check whether the ice cream is fresh*). For each version there is a tag `WCS_Capabilities` with attribute version, e,.g.

```
<WCS_Capabilities version="1.0.0">
...
</WCS_Capabilities>
```

For each version a number of datasets is hosted. Select one dataset (ice cream flavour) from the list through the `ContentMetadata` tag. The name of the dataset is the `name` tag, in this case there is only `topo`.

```
<ContentMetadata>
<CoverageOfferingBrief>
<description>topo meters true  Topography</description>
<name>topo</name>
<label>Topography</label>
...
</CoverageOfferingBrief>
</ContentMetadata>
```

Request some content of a WCS server

Now we can actually get a subset from the dataset want by using `request=GetCoverage` instead of the `request=GetCapabilities` we used above to obtain the meta-data (*order ice cream*). The following <keyword,value> pairs are mandatory.

keyword	value	source
service	WCS	Mandatory WCS standard value
request	GetCoverage	Mandatory WCS standard value
version	1.0.0	One of the mandatory WCS standard values returned by returned by the `request=GetCapabilities`
BBOX	0,50,10,55	bounding box: min(longitude),min(latitude),max(longitude),max(latitude)
coverage	topo	one of the `ContentMetadata name` returned by `request=GetCapabilities`. WCS counterpart of WMS `layers`
format	`GeoTIFF`,`NetCDF3`,`NetCDF4`,`AAIGRID`	Unlike `WMS`, the available formats (encoding types) are not returned by `request=GetCapabilities`, but by an extra call with `request=DescribeCoverage`. Well-known formats for WCS servers are: `GeoTIFF`,`NetCDF3`,`NetCDF4` and `AAIGRID` (Arc Ascii Grid).

These keywordd return data in the native resolution, some servers demand specification of a target grid:

keyword	value	source
crs	crs:84	coordinate system
resx	0.1	resolution
resy	0.1	resolution

Unlike WMS, to known which encoding formats are available, there is a separate request value `DescribeCoverage`
`http://geoport.whoi.edu/thredds/wcs/bathy/srtm30plus_v6?service=WCS&request=DescribeCoverage&version=1.0.0`
which is available when the `request=GetCapabilities` contains

```
<Capability>
...
<DescribeCoverage>
...
<OnlineResource xlink:href="http://geoport.whoi.edu/thredds/wcs/bathy/srtm30plus_v6" />
</DescribeCoverage>
..
</Capability>
```

In this case `request=DescribeCoverage` offers three formats

```
...
<supportedFormats>
<formats>GeoTIFF</formats>
<formats>GeoTIFF_Float</formats>
<formats>NetCDF3</formats>
</supportedFormats>
...
```

For 2.0.1 request this is:

2.0.1 Describe Coverage

```
https://rwsprojectarchief.openearth.nl/geoserver/mepduinen/wcs?service=WCS&version=2.0.1
&request=DescribeCoverage&CoverageId=SpanjaardsDuinT0_50cm
```

Important parameters for subsetting are described in the output. Check for srsName and axisLabels. These need to be used in the getCoverage request.

Summary

correct WCS request for two available formats are given below. You can copy these into your browser to obtain the data. We chose the same bounding boxes as in the OPeNDAP primer that access these same datasets via netCDF libraries in Matlab, python or R. We discussed the pros and cons of WCS vs OPeNDAP in a FOSS4G paper.

```
http://geoport.whoi.edu/thredds/wcs/bathy/srtm30plus_v6?request=GetCoverage
&version=1.0.0
&service=WCS
&format=geotiff
&coverage=topo
&BBOX=0,50,10,55
```

```
http://geoport.whoi.edu/thredds/wcs/bathy/srtm30plus_v6?request=GetCoverage
&version=1.0.0
&service=WCS
&format=netcdf3
&coverage=topo
&BBOX=0,50,10,55
```

for which the netCDF file look like

```
NetCDF-3 Classic srtm30plus_v6_netcdf3.nc {


dimensions:
        lat = 601 ;
        lon = 1201 ;

variables:
        int16 topo(lat,lon), shape = [601 1201]
                topo:_FillValue = -9999 s
                topo:grid_mapping = "GDAL_Geographics"
                topo:long_name = "Topography"
                topo:units = "meters"
                topo:coordinates = "lat lon lat lon"
        double lat(lat), shape = [601]
                lat:units = "degrees_north"
                lat:long_name = "Latitude"
                lat:_CoordinateAxisType = "Lat"
                lat:standard_name = "latitude"
        double lon(lon), shape = [1201]
                lon:units = "degrees_east"
                lon:long_name = "Longitude"
                lon:_CoordinateAxisType = "Lon"
                lon:standard_name = "longitude"

//global Attributes:
                :Conventions = "CF-1.0"
                :AREA_OR_POINT = "Area"
                :title = "UCSD SRTM30_v6 Global DEM (30 sec)"
                :History = "Translated to CF-1.0 Conventions by Netcdf-Java CDM (NetcdfCFWriter)
Original Dataset = bathy/srtm30plus_v6; Translation Date = Tue Nov 08 05:16:29 EST 2011"

}
```

You can readily open and visualize this file with for instance ncBrowse.

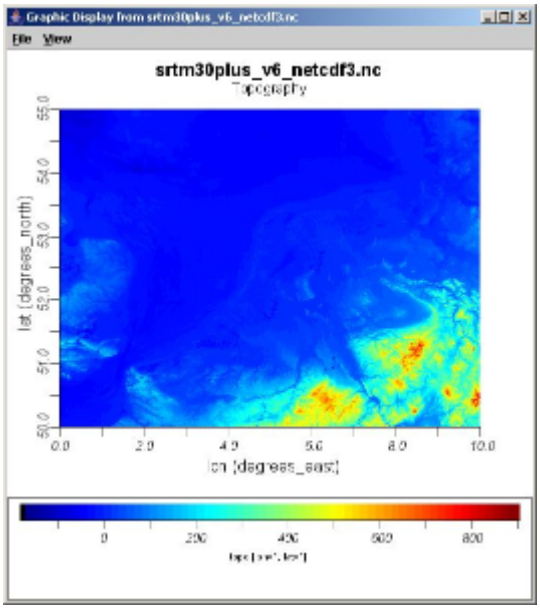
Some additional, optional WCS keywords are:

keyword	value	source
time	yyyy-mm-ddTHH:MM:SSZ	ISO time notation. Not implemented in most GIS minded software packages, for an example implementation see the ADAGUC WMS server and WMS web client
RESX	numeric value	interpolation resolution in x direction within BBOX, if supported: in the example above the original data points are returned.
RESY	numeric value	interpolation resolution in y direction within BBOX, if supported: in the example above the original data points are returned.

For 2.0.1 the full GetCoverage request is presented below (incl. subsetting)

2.0.1 subset GetCoverage request

```
https://rwsprojectarchief.openearth.nl/geoserver/mepduinen/wcs?service=WCS
&version=2.0.1
&CoverageId=SpanjaardsDuinT0_50cm
&request=GetCoverage
&format=image/tiff
&subset=Y(446730.5,447414.5)
&subset=X(68628.5,68998.5)
```



Accessing WCS by Python OWSLib library

Similar to WFS and WMS accessibility OWSLib has the ability to get data from a WCS service. The code below is a start and gives the contents of data accessible as a WCS.

```
import os
import tempfile
from owslib.wcs import WebCoverageService
url = 'http://deltaresdata.openearth.nl/geoserver/ows?'
wcs = WebCoverageService(url,version='1.0.0')
wcs.identification.type
wcs.identification.title
list(wcs.contents)
```

The code above results in a list of available rasters. For Sediment Thickness of the world some characteristics are retrieved via next code.

```
sed = wcs['global:Sediment Thickness of the World']
sed.keywords
sed.grid.highlimits
sed.boundingboxes
```

WCS enables you to get a specific part for a specific bounding box like in the following example. For a specified bounding box a tif will be written to a temporary file

```
requestbbox = (-6.283406057098745, 36.59251369598765, -6.266739197530853, 36.6050139853395)
requestwidth = 4
requestheight = 3
gc = wcs.getCoverage(identifier=sed.id,
                                  bbox=requestbbox,
                                  format='GeoTIFF',
                                  crs=sed.boundingboxes[0]['nativeSrs'],
                                  width=requestwidth,
                                  height=requestheight)
# random unique filename
tmpdir = tempfile.gettempdir()
fn = os.path.join(tmpdir,'test.tif')
f = open(fn,'wb')
f.write(gc.read())
gc.close()
f.close()
```