NARLabs

Overview and Framework of Taiwan Data Cube

Charlotte Hsu

National Space Organization, NarLabs

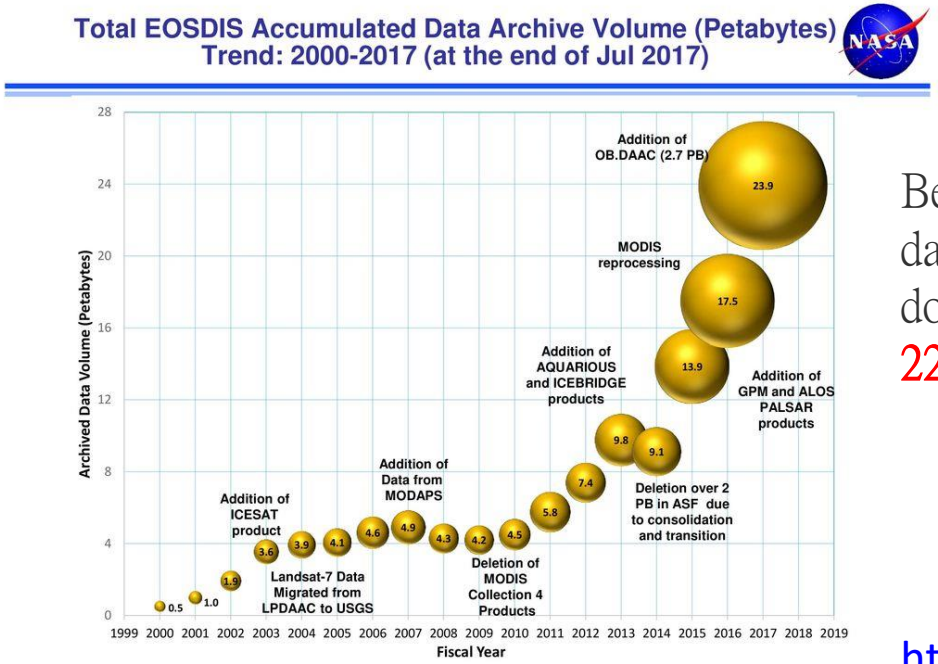
Feb. 8th 2021

AGENDA

1. Introduction
2. Overview
3. The structure of the TWDC
   a. Data Cube Ingestion
   b. Data Cube web-based User Interface (UI)
   c. Jupyter Notebook Python Tutorial

Introduction

- The free and open policy adopted by the European Space Agency (ESA) and NASA and the U.S. Geological Survey (USGS) has driven exponential growth in users downloading Earth observation data.

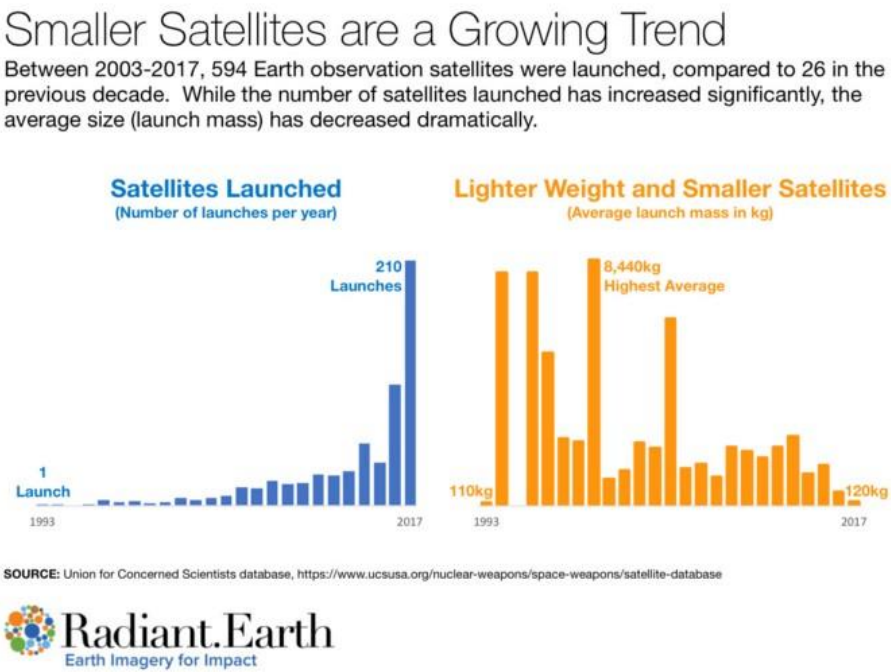


Between 2015 and 2017, the volume of data in the EOSDIS archive more than doubled, from roughly 10 PB to almost 22 PB (see graphic).

https://slideplayer.com/slide/13140310/

Introduction

- New generations of Earth Observation(EO) satellites are creating increasingly significant volumes of data with such comprehensive global coverage that for many applications.

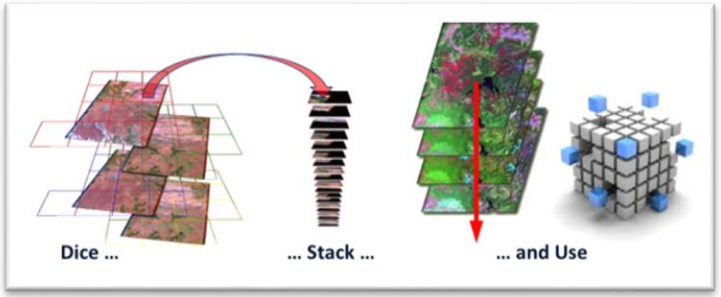


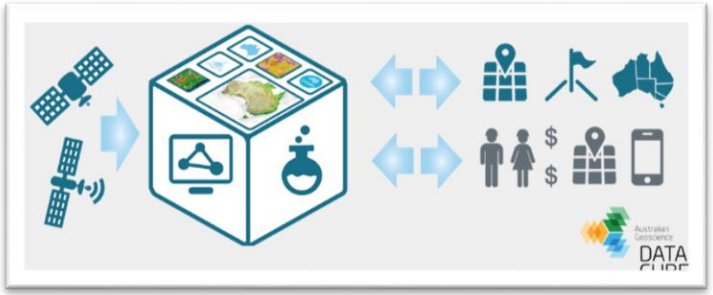
The lack of data is no longer a limiting factor.

https://medium.com/radiant-earth-insights/observing-the-earth-fueling-global-development-solutions-1c69fd5632bc

Introduction

- The data management and analysis challenges arising from the huge increase in free and open data volumes can be overcome with new computing infrastructures, technologies and data architectures, such as the Open Data Cube.





Introduction

- Your data (may be > TB) is in your remote cluster.
- You don't want to download your data.
- You want to access and interactively play with your data in Jupyter notebook.

https://images.app.goo.gl/g9yxudEazYNEVawAA

https://images.app.goo.gl/2YYMysuQSHq17yTJ7

Overview

- The Open Data Cube (ODC) is an Open Source Geospatial Data Management & Analysis project that helps you harness the power of Satellite data.
- The ODC is a set of Python libraries and PostgreSQL database that helps you work with geospatial raster data.

Overview

- The ODC core serves as a layer between satellite data and end user applications.
- It provides a common analytical framework to allow multiple data sources to produce information for multiple uses.

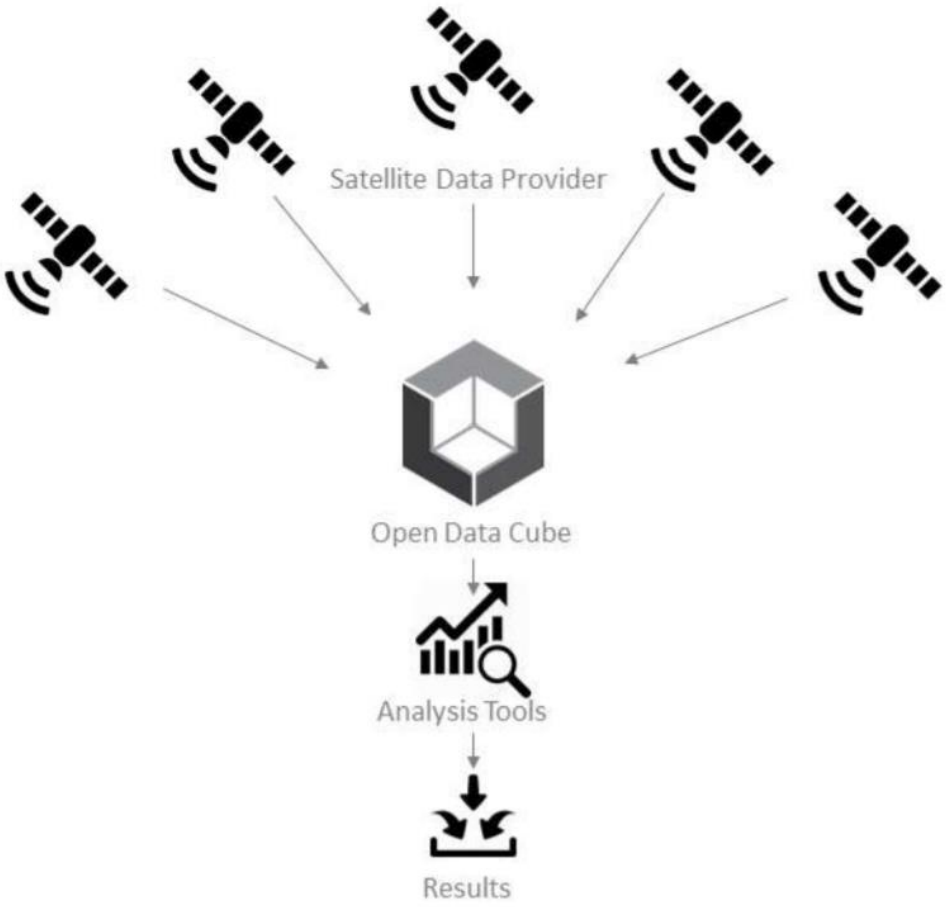
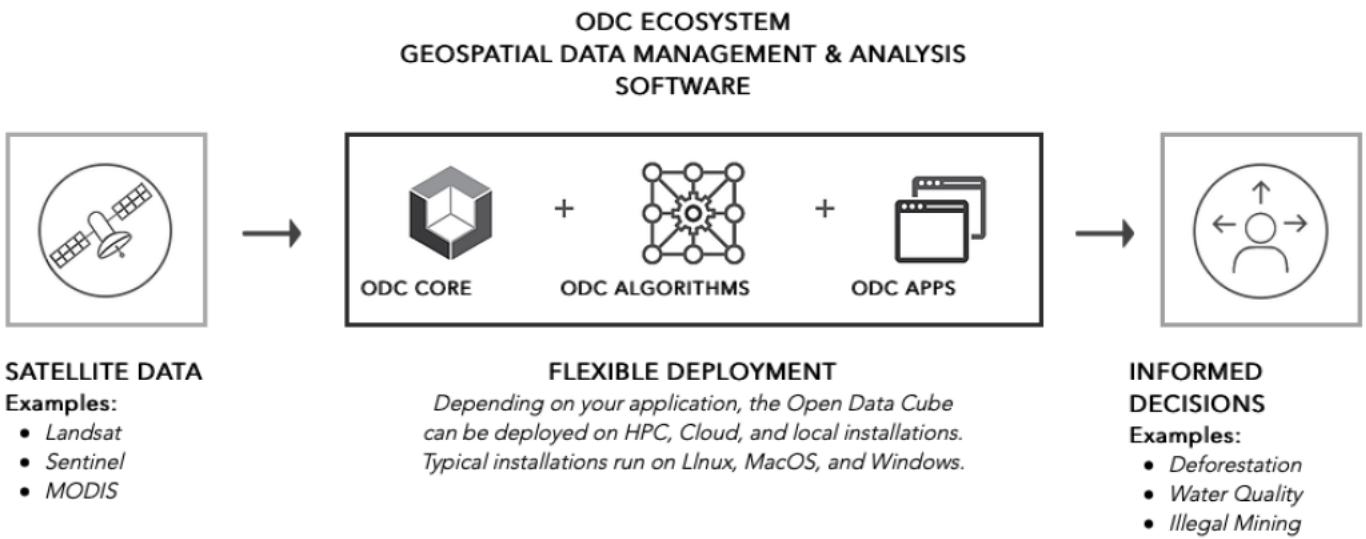


Figure 1: High-Level Open Data Cube Overview

Overview

- The ODC was developed for the analysis of temporally-rich earth observation data.
- Such data may include elevation models, geophysical grids, interpolated surfaces and model outputs.

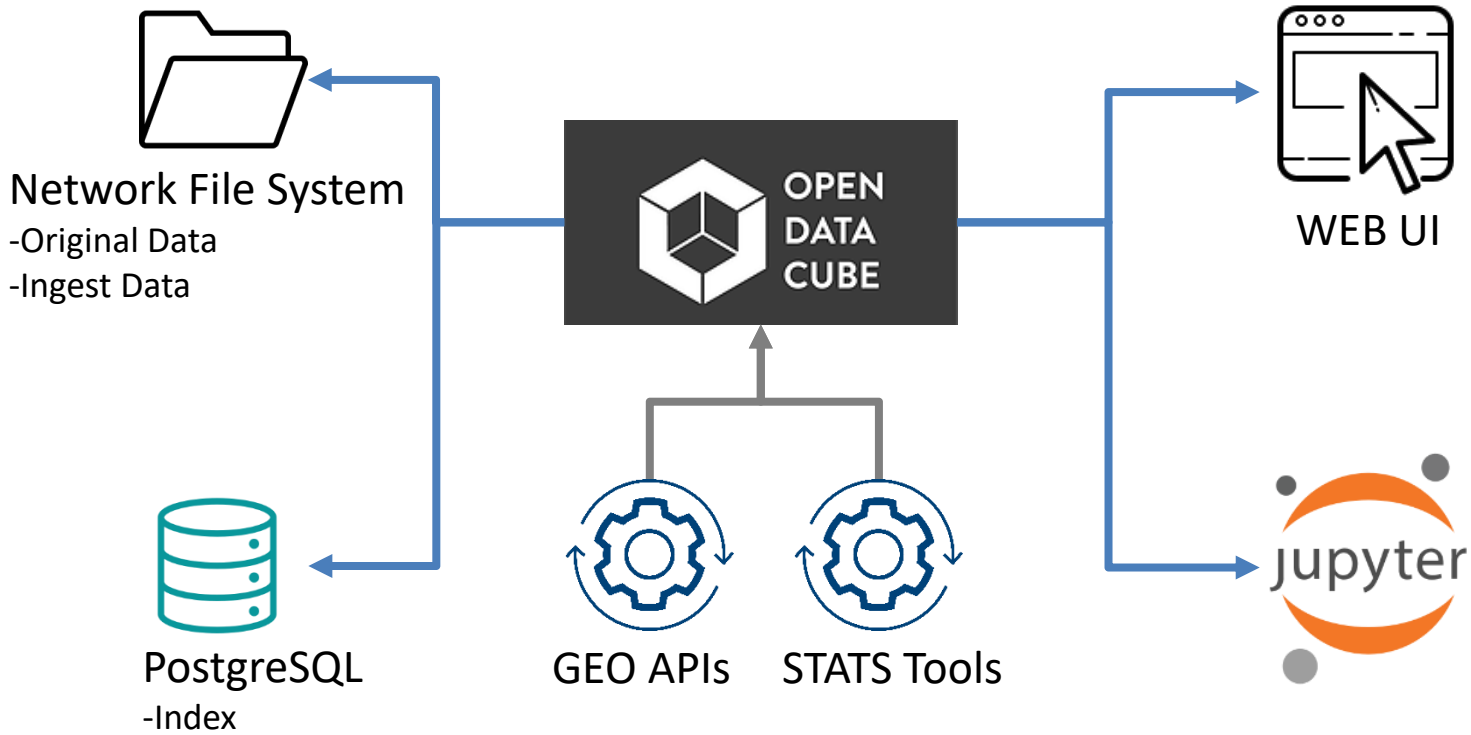


Overview

- The Data Cube works well with Analysis Ready Data (ARD), pre-processed, ready to use data made available by data providers.
- The installation of building a Data Cube environment that can be used to ingest data and run analytics processes.

The structure of TWDC

- data, an index and software



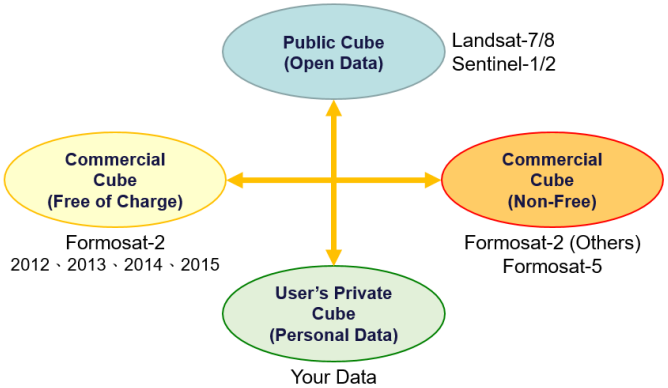
https://medium.com/opendatacube/what-is-open-data-cube-805af60820d7

The structure of TWDC

Taiwan Data Cube

+

TWDC
衛星多元遙測資料服務平台



Advantages:

1. Store and manage multiple satellites data in different databases
2. Private Dara Cubes derive from TWCC management
3. Multiple Cubes Data Sharing: private Cubes access data from other Cubes
4. High-speed computation due to powerful supercomputer

The structure of TWDC

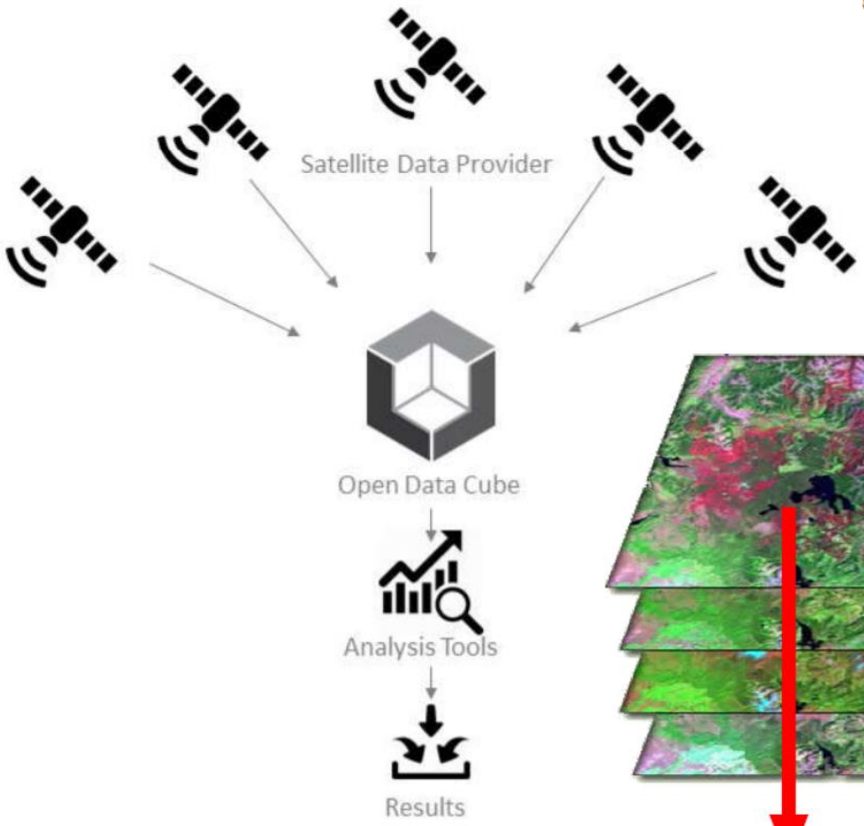
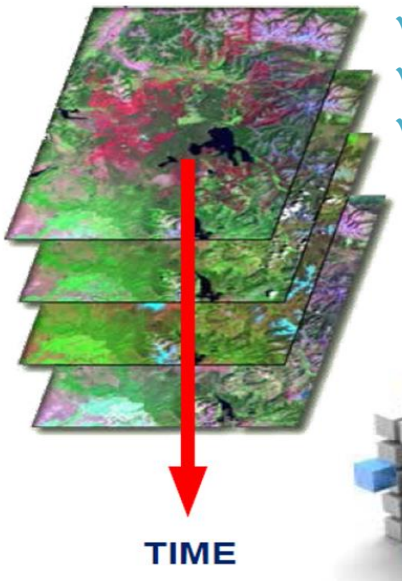


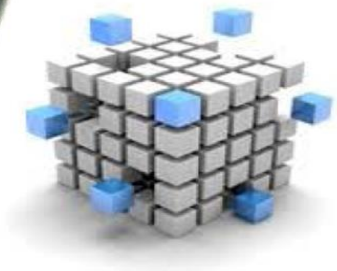
Figure 1: High-Level Open Data Cube Overvi



Integration of multiple satellites data

- ✓ FS5(By Order)
- ✓ FS2(By Order)
- ✓ Landsat7
- ✓ Landsat8
- ✓ Sentinal-1
- ✓ Sentinal-2

others



ARD ingestion

- ✓ Analysis ready data
- ✓ Ingestion

Time series comparison

- ✓ Multi-sequential data query
- ✓ Multiple satellites data query

Multidimensional data analysis

- ✓ Data analysis
- ✓ Results comparison

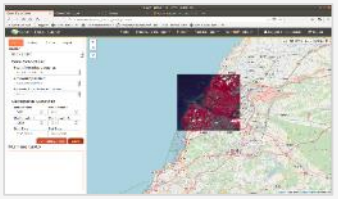
The structure of TWDC

Data management and query

- Query
- Data type management
- Data modification

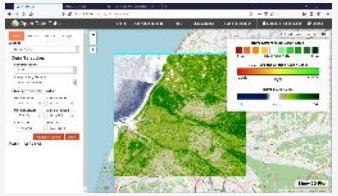
Data collection and output

- Data images output
- Complex image data (optional)
- Multi-sequential attributes query



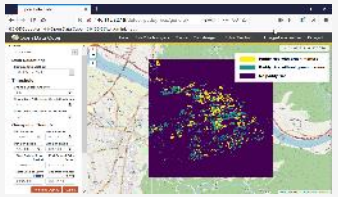
Analyses

- Spectrum index analyses
  - NDVI, NDWI, NDBI, ...



Customized applications

- Customized analyses
  - WEB UI Development
  - Python Scripting

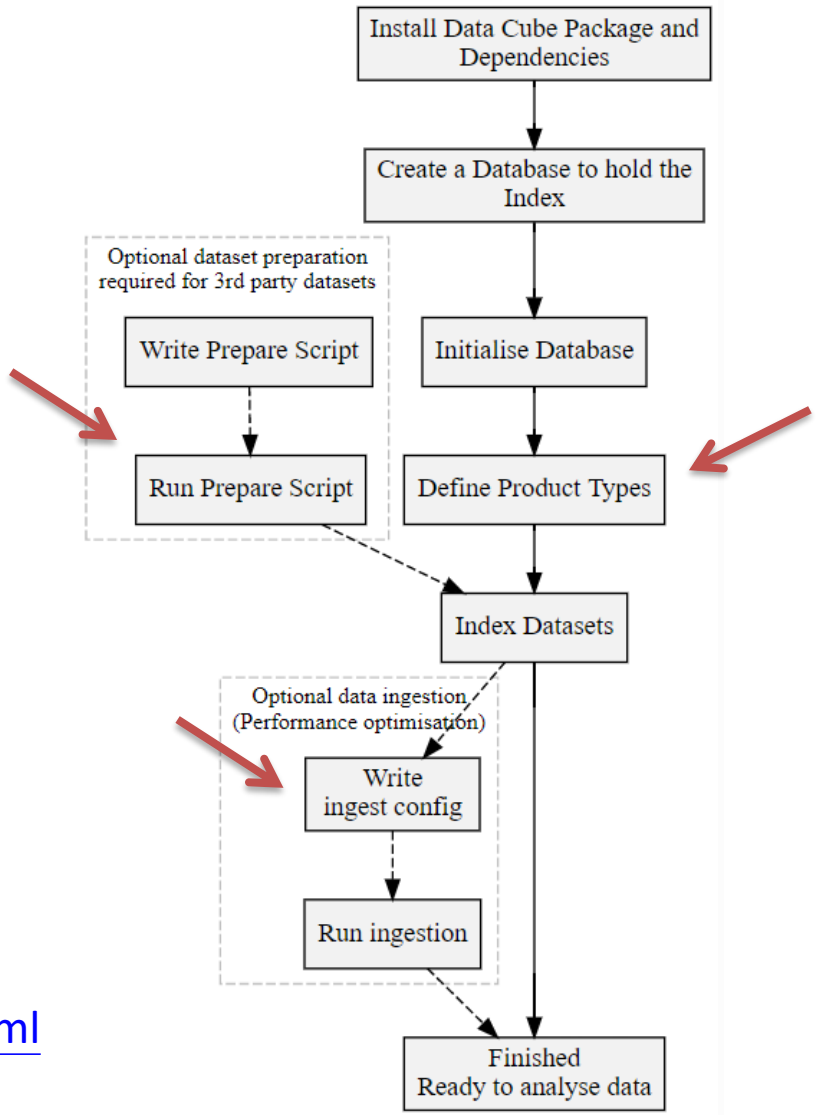


Data Cube Ingestion

Data Cube Ingestion

- **Data** is usually file based, either in local directories of GeoTIFFs or NetCDF files.
- **Index**, the ODC uses PostgreSQL to store a list of Products and Datasets. User without need to know where the required files are stored.

https://datacube-core.readthedocs.io/en/latest/ops/overview.html



Data Cube Ingestion

- What is ingestion?
  - Ingestion is the entire process of adding new data to the Data Cube.
- This process includes:
  - Describing your source dataset in a well-defined schema(.yaml)
  - Creating a script for each dataset to generate a metadata file(.yaml).
  - Creating an ingestion configuration file that defines the input dataset type and the output characteristics.

Product Definition

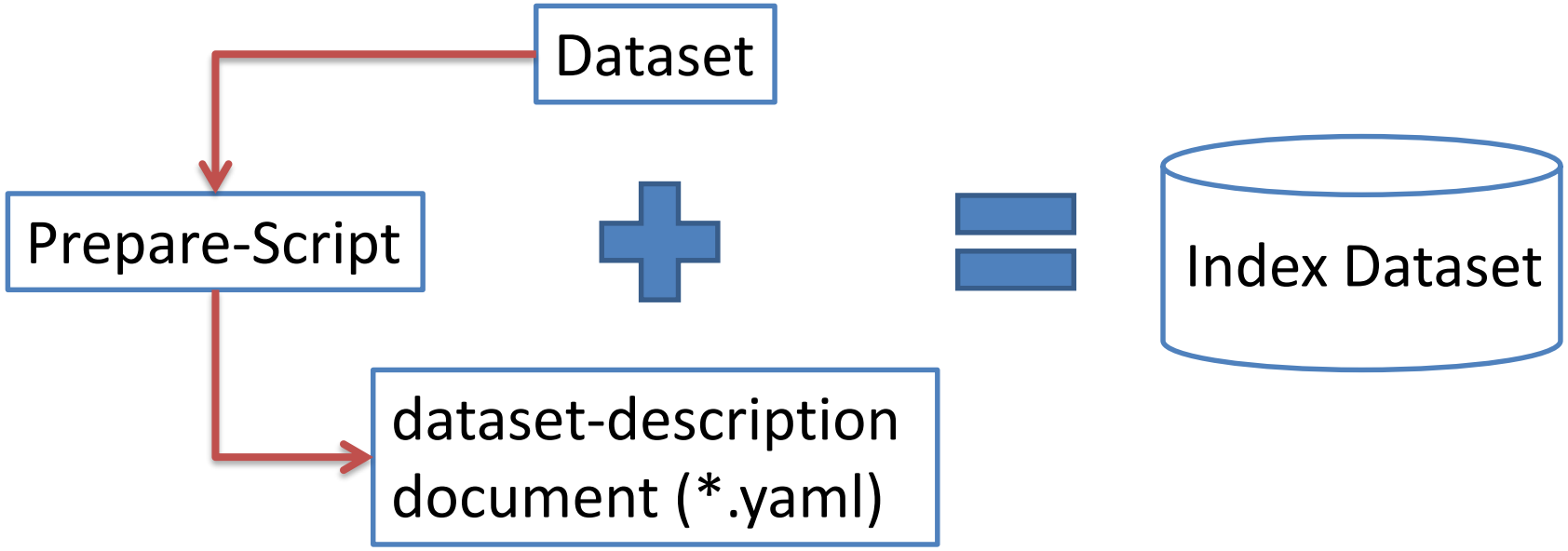
- defines some of the metadata common to all the datasets.
- $ datacube product add product.yaml

```yaml
name: dsm1sv10
description: DSM 1sec Version 1.0
metadata_type: eo

metadata:
    platform:
        code: SRTM
    instrument:
        name: SIR
    product_type: DEM
    format:
        name: ENVI

storage:
    crs: EPSG:4326
    resolution:
        longitude: 0.000277777777780
        latitude: -0.000277777777780

measurements:
      - name: elevation
        dtype: float32
        nodata: .nan
        units: 'metre'
```

https://datacube-core.readthedocs.io/en/stable/ops/product.html

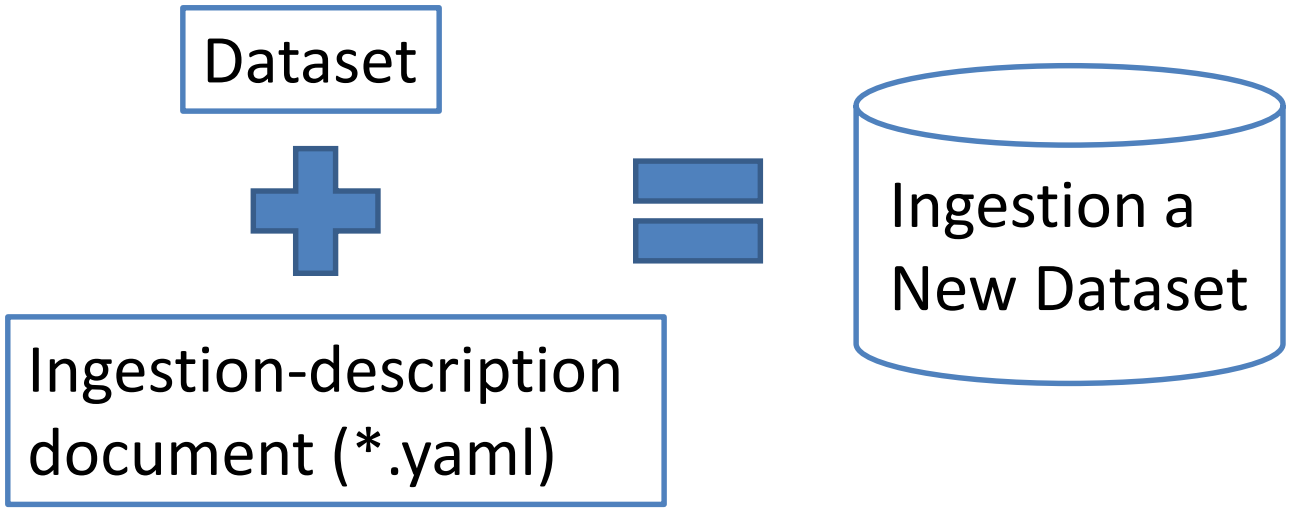
Dataset Preparation Scripts & Indexing

- Some data you may want to load into your Data Cube will come pre-packaged with a dataset-description document.
- $ datacube dataset add dataset-description.yaml



Ingestion

- Ingestion configuration file
  - defining a transformation between the source dataset and the output dataset.
  - Using the ingestion process to define a bounding box, a resolution, projection and tile size.
- $ datacube ingest -c ingestion-description.yaml



Data Cube web-based User Interface (UI)

Data Cube Manager

Taiwan Data Cube

Home | Data Cube Manager | Tools | Task Manager | Submit Feedback | Log In

- Data Cube Visualization
- Dataset Types
- Dataset Viewer
- Ingestion Configuration Builder
- Ingestion on Demand

Welcome to the Taiwan Data Cube(TWDC)

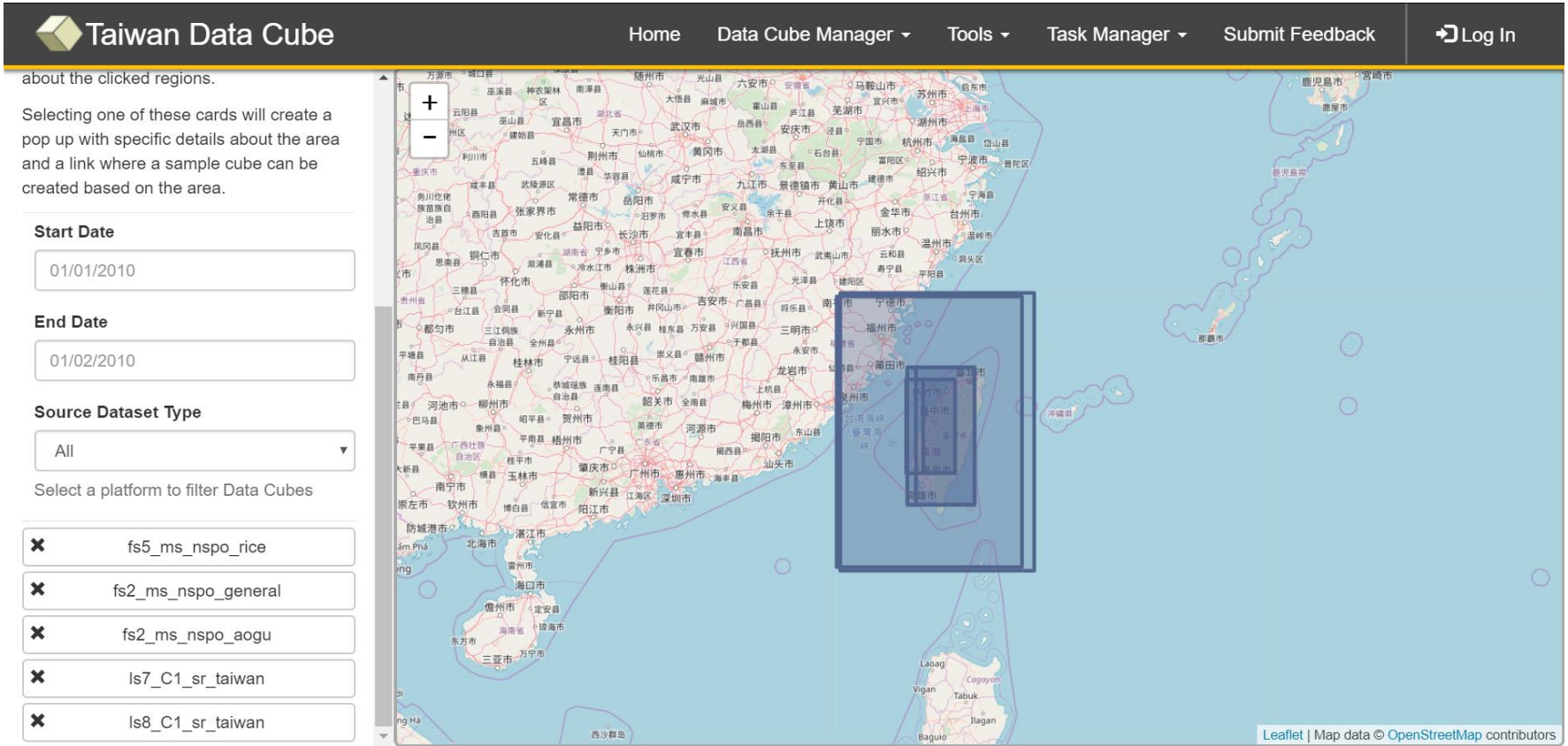
CEOS is using the power of the Open Data Cube to help address the needs of satellite data users, giving them a better picture of their land resources and land change.

- Ease of use and access to satellite-based data
- Multiple dataset interoperability and spatial consistency
- Use of "Analysis Ready" Data Products
- A Shift in Paradigm from Scenes to Pixels

Log In

Visualize Data Holdings

- Explore the Data Cube Manager > Data Cube Visualization menu to view the location and size of available data cubes.



Dataset Types

- Datasets in the Data Cube are organized using a data-type system.

Taiwan Data Cube

Home | Data Cube Manager | Tools | Task Manager | Submit Feedback | Log In

Dataset Types

Dataset type definitions are used to describe datasets contained in the Data Cube. Individual datasets are associated with a single dataset type. Each dataset type includes a variety of data and metadata including dataset measurement data, product types and platforms, and creation dates and users.

Show 10 entries

Search:

Id	Name	Platform	Instrument	Product Type	Measurements	Description	View Datasets	View Full Definition
1	fs2_ms_nspo_toa_scene	FS2_MS_NSPO	R1M	NSPOortho	TOA_band3, TOA_band2, TOA_band1, TOA_band4...	FORMOSAT 2 MS NSPO TOA scene	View datasets	View definition
2	fs2_ms_nspo_general	FS2_MS_NSPO	R1M	NSPOortho	blue, green, red, nir, pixel_qa,	FS2 proessed by NSPO Resampled to 8m EPSG:4...	View datasets	View definition
3	fs2_ms_nspo_masadi	FS2_MS_NSPO	R1M	NSPOortho	blue, green, red, nir, pixel_qa,	FS2 proessed by NSPO Resampled to 8m EPSG:4...	View datasets	View definition
4	fs2_ms_nspo_aogu	FS2_MS_NSPO	R1M	NSPOortho	blue, green, red, nir, pixel_qa,	FS2 proessed by NSPO Resampled to 8m EPSG:4...	View datasets	View definition
5	fs2_ms_nspo_thailand	FS2_MS_NSPO	R1M	NSPOortho	blue, green, red, nir, pixel_qa,	FS2 proessed by NSPO Resampled to 8m EPSG:4...	View datasets	View definition
6	fs5_ms_nspo_toa_scene	FS5_MS_NSPO	R1M	NSPOortho	TOA_band3, TOA_band2, TOA_band1, TOA_band4...	FORMOSAT 5 MS NSPO TOA scene	View datasets	View definition
7	fs5_ms_nspo_general	FS5_MS_NSPO	R1M	NSPOortho	blue, green, red, nir, pixel_qa,	FS5 proessed by NSPO Resampled to 4m EPSG:4...	View datasets	View definition
8	ls7_C1_sr_scene	LANDSAT_7	ETM	LEDAPS	blue, green, red, nir, swir1, swir2, sr_atmos_opacity,...	Landsat 7 USGS Collection 1 Level2 Surface Reflec...	View datasets	View definition
9	ls7_C1_sr_taiwan	LANDSAT_7	ETM	LEDAPS	blue, green, red, nir, swir1, swir2, sr_atmos_opacity,...	Landsat 7 USGS Collection 1 Level2 Surface Reflec...	View datasets	View definition
10	ls8_C1_sr_scene	LANDSAT_8	OLI_TIRS	LaSRC	coastal_aerosol, blue, green, red, nir, swir1, swir2, p...	Landsat 8 USGS Collection 1 Level2 Surface Reflec...	View datasets	View definition

https://twdc.colife.org.tw

Previous 1 2 Next

Dataset Viewer

- The dataset viewer panel organizes all individual products in the Data Cube.

Taiwan Data Cube | Home | Data Cube Manager | Tools | Task Manager | Submit Feedback | Log In

Datasets

Datasets are individual chunks of data - either full acquisitions (scenes) or chunks of acquisitions created during ingestion. Information found in the dataset is the id, added date, and acquisition/chunk data such as extents, date acquired, and path to dataset. The table found on this page contains the datasets that match the criteria specified by the datasets form.

Products

fs2_ms_nspo_toa_scene - Source only
fs2_ms_nspo_general - Ingested only
fs2_ms_nspo_masadi - Ingested only
fs2_ms_nspo_aogu - Ingested only

Min Latitude

Min Latitude

Max Latitude

Max Latitude

Min Longitude

Min Longitude

Max Longitude

Max Longitude

Start Date

01/01/2010

End Date

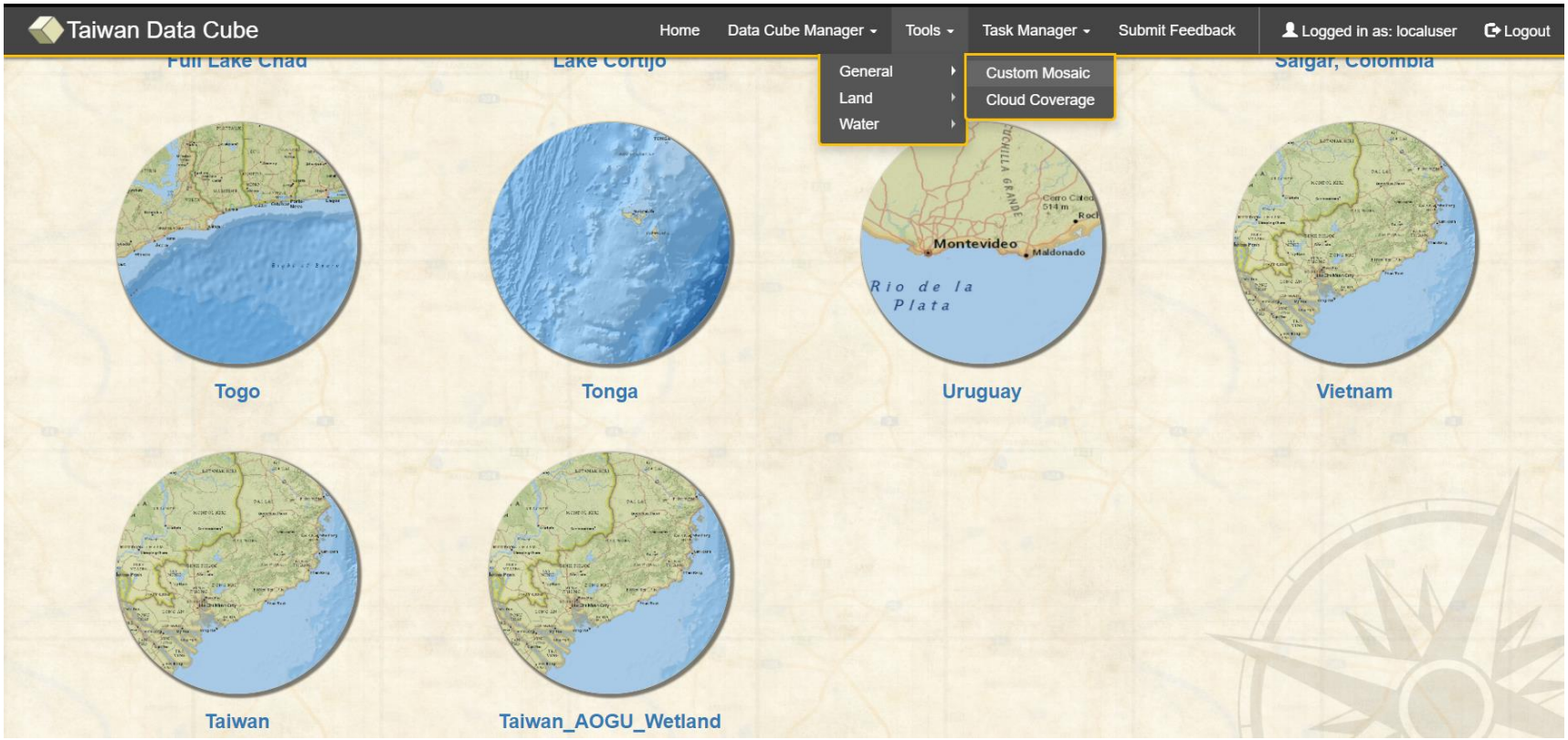
01/02/2010

Show 15 entries

Id	Platform	Instrument	Product Type	Upper Left	Lower Right	Dataset Acq. Date	Format
cf42e36a-33d5-43ca-aa3d-0aac353175f2	FS2_MS_NSPO	R1M	NSPOortho	100.21, 14.43	100.73, 14.12	2015-12-30 03:48:40	GeoTiff
66ea293f-6ea3-4c22-8ed4-03f693617e7a	FS2_MS_NSPO	R1M	NSPOortho	100.29, 14.42	100.79, 14.12	2015-05-18 03:49:30	GeoTiff
95a32ef4-23ec-4cf1-9a93-1b1f681fd724	FS2_MS_NSPO	R1M	NSPOortho	100.40, 14.40	100.89, 14.10	2011-10-23 03:45:17	GeoTiff
e9abdd51-1954-4f8f-85ee-be391f79a5f6	FS2_MS_NSPO	R1M	NSPOortho	100.30, 14.42	100.78, 14.12	2010-02-25 03:33:23	GeoTiff
8a63fd9e-b675-4d7c-931a-f99825901900	FS2_MS_NSPO	R1M	NSPOortho	120.16, 22.41	120.46, 22.16	2016-02-13 02:04:06	GeoTiff
42540969-7864-48f6-aae5-d0b927c9aa51	FS2_MS_NSPO	R1M	NSPOortho	120.16, 22.41	120.46, 22.16	2016-02-09 02:03:58	GeoTiff
1f9338b0-c9ea-441d-8800-d86f245190e4	FS2_MS_NSPO	R1M	NSPOortho	120.16, 22.41	120.46, 22.16	2016-01-26 02:04:04	GeoTiff
d2547382-de92-47ef-bd83-f2bcc6b8d63c	FS2_MS_NSPO	R1M	NSPOortho	120.17, 22.42	120.47, 22.16	2015-02-01 02:05:42	GeoTiff
e5cd9570-84d0-46b8-9753-27206509e94f	FS2_MS_NSPO	R1M	NSPOortho	120.17, 22.41	120.47, 22.16	2015-01-01 02:05:40	GeoTiff
beacffcc-50f0-4534-9f2c-9415e390cc93	FS2_MS_NSPO	R1M	NSPOortho	120.20, 22.61	120.51, 22.36	2016-02-13 02:04:02	GeoTiff
e5965de5-9715-4e31-b97b-bf0eac1317e0	FS2_MS_NSPO	R1M	NSPOortho	120.20, 22.61	120.51, 22.36	2016-02-09 02:03:54	GeoTiff
70203715-7f33-448d-8b09-cb0b6417d6eb	FS2_MS_NSPO	R1M	NSPOortho	120.21, 22.61	120.51, 22.36	2016-01-26 02:04:00	GeoTiff
8b205ba7-cae7-486f-b0df-aeb426232891	FS2_MS_NSPO	R1M	NSPOortho	120.21, 22.61	120.52, 22.36	2015-02-01 02:05:38	GeoTiff

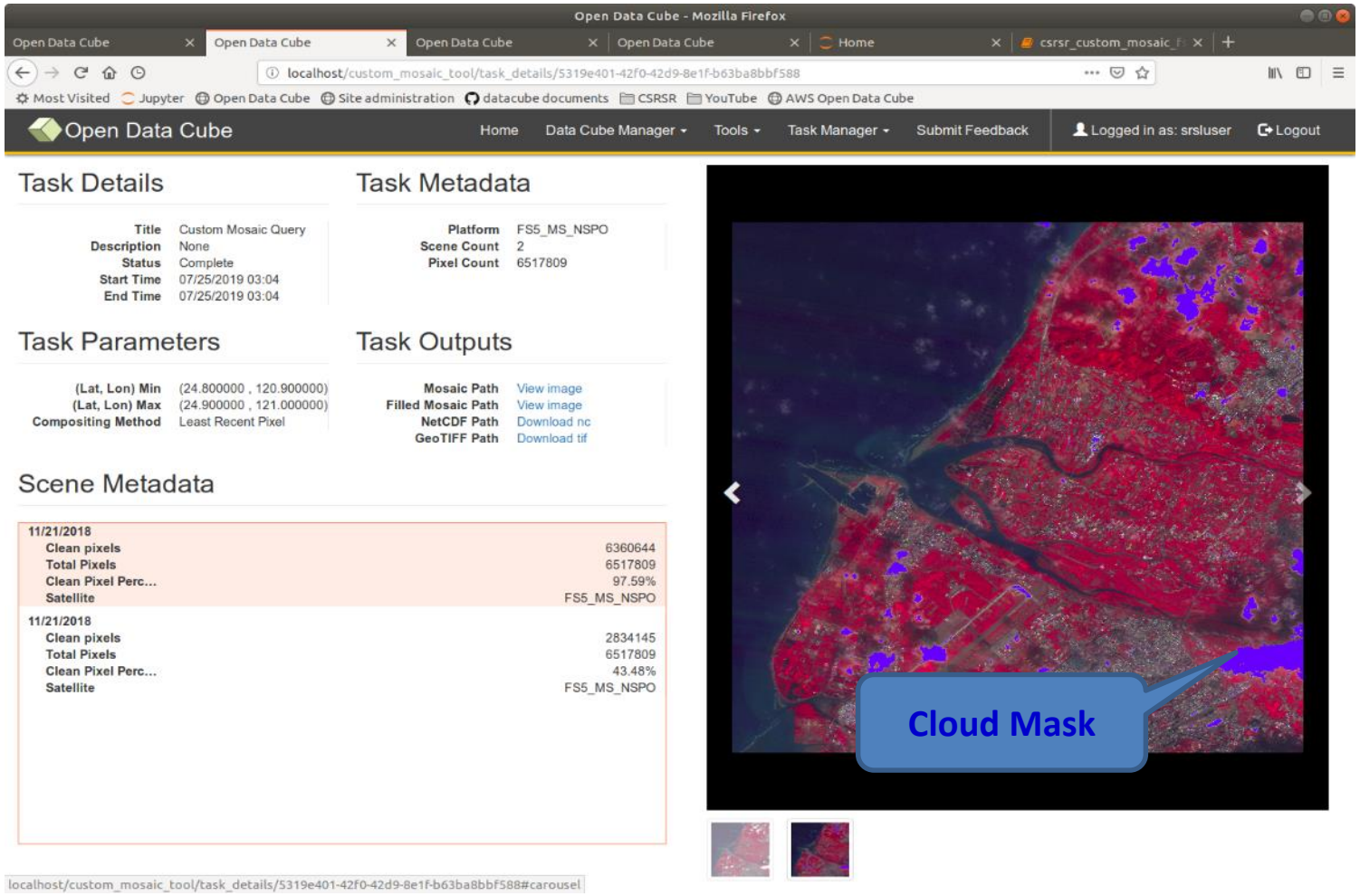
Custom Mosaic Tool

- Select Custom Mosaic Tool. This will give you a list of data cubes. Pick a cube of interest.



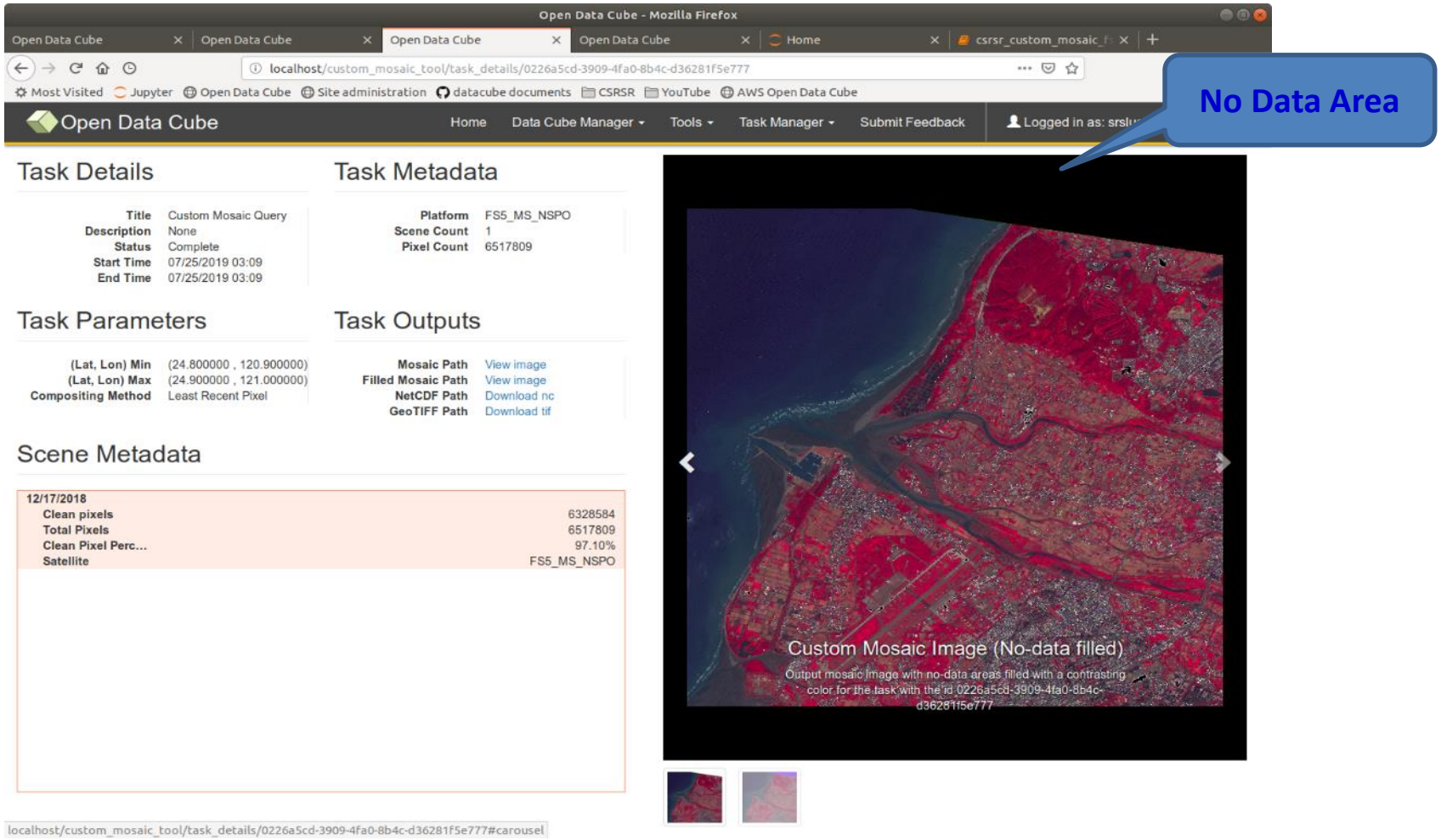
Custom Mosaic Tool

Date: Nov. 21st 2018



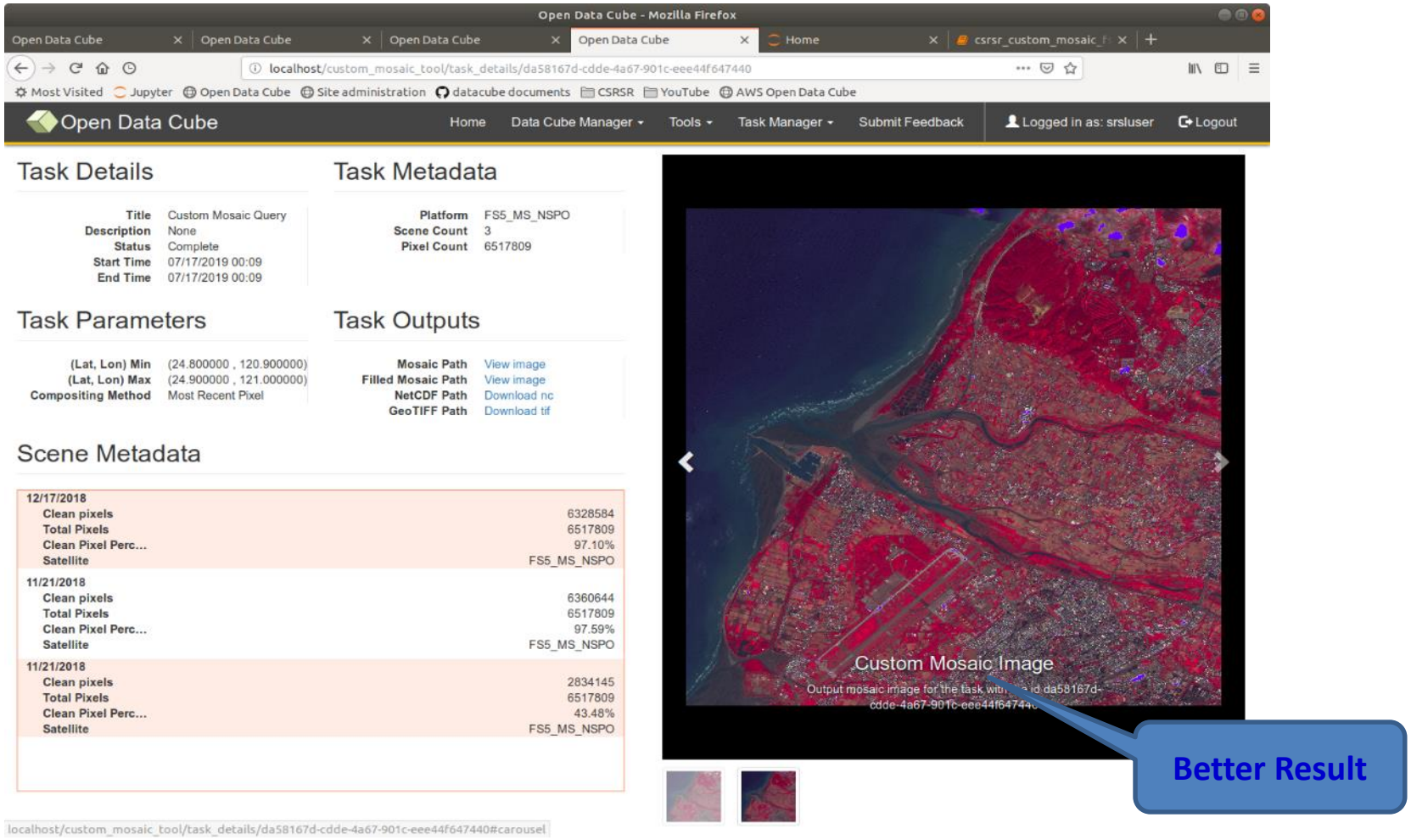
Custom Mosaic Tool

Dec. 17th 2018



Custom Mosaic Tool

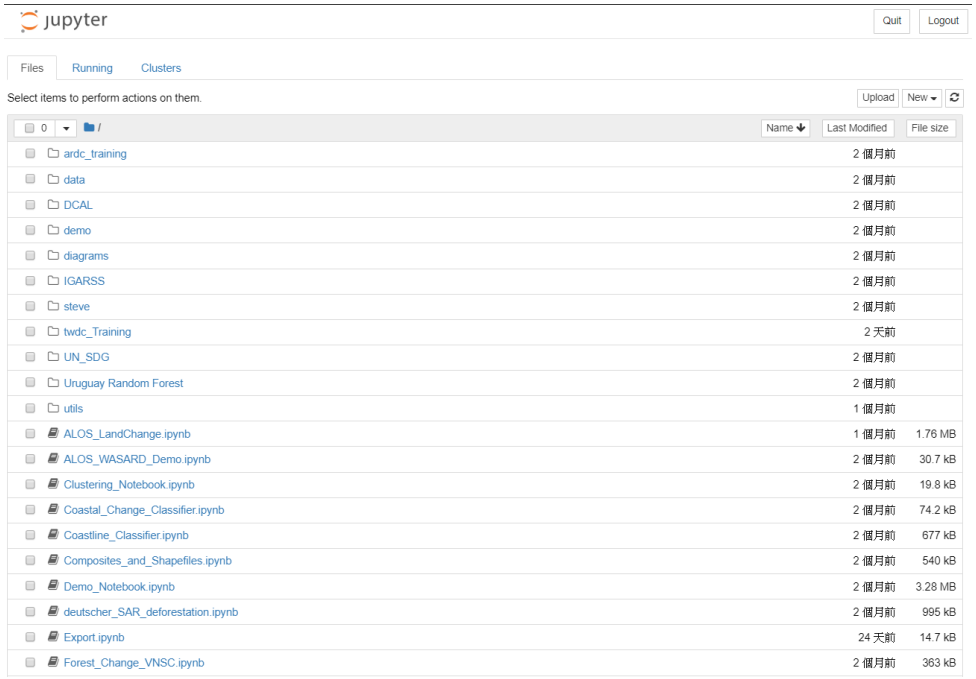
After custom mosaic



Jupyter Notebook

What is Jupyter?

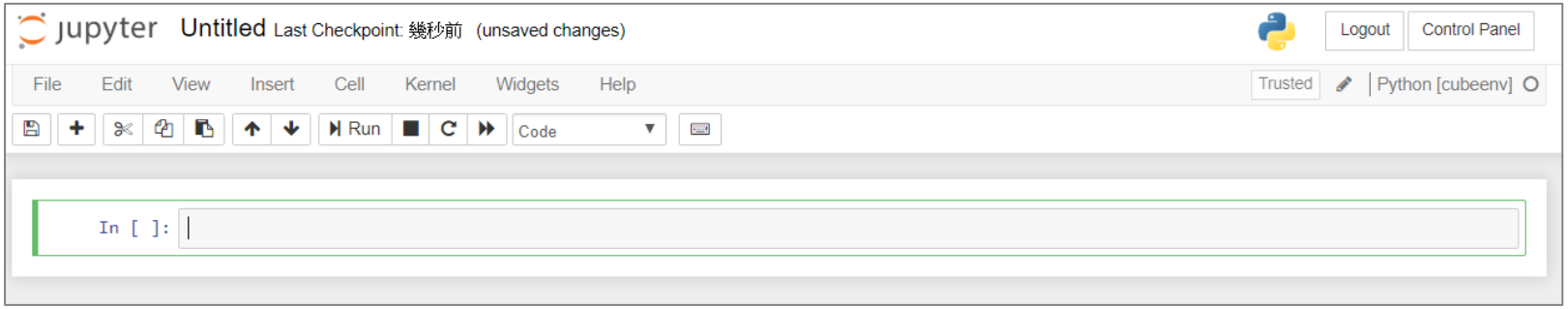
- Jupyter notebook is a web application that will let the user create documents containing code and documentation, such as equations, plots and text.



What is Jupyter?

- click "New" to create a new notebook

- Type the command in each cell and then hit Shift-Enter to run and to create new Cell.



Jupyter Notebook Python Tutorial

Python DataCube

- Import the Datacube configuration

Creating the datacube object

In [1]:
```
import os,sys
from datacube.config import LocalConfig
```

In [2]:
```
import datacube
dc = datacube.Datacube(app = 'my_app')
```

List all products of datacube

In [3]:
```
list_of_products = dc.list_products()
netCDF_products = list_of_products[list_of_products['format'] == 'NetCDF']
netCDF_products
```

Out[3]:

id	name	description	creation_time	platform	time	instrument	lon	label	lat	product_type	format	crs	reso
4	fs2_ms_nspo_aogu	FS2 proessed by NSPO Resampled to 8m EPSG:4326...	None	FS2_MS_NSPO	None	R1M	None	None	None	NSPOortho	NetCDF	EPSG:4326	(-7.836 7.836
2	fs2_ms_nspo_general	FS2 proessed by NSPO Resampled to 8m EPSG:4326...	None	FS2_MS_NSPO	None	R1M	None	None	None	NSPOortho	NetCDF	EPSG:4326	(-7.836 7.836

Python DataCube

Pick a product

Use the platform and product names from the previous block to select a Data Cube.

In [4]:
```
import utils.data_cube_utilities.data_access_api as dc_api
api = dc_api.DataAccessApi()

# Change the data platform and data cube here

platform = "LANDSAT_7"
#platform = "LANDSAT_8"
#platform = "FS2_MS_NSPO"
#platform = "FS5_MS_NSPO"

product = "ls7_C1_sr_taiwan"
#product = "fs2_ms_nspo_aogu"
#product = "fs5_ms_nspo_rice"

# Get Coordinates
coordinates = api.get_full_dataset_extent(platform = platform, product = product)
```

Display Latitude-Longitude and Time Bounds of the Data Cube

In [5]:
```
latitude_extents = (min(coordinates['latitude'].values),max(coordinates['latitude'].values))
print( latitude_extents )
```

```
(20.684383153326095, 27.001875220428406)
```

In [6]:
```
longitude_extents = (min(coordinates['longitude'].values),max(coordinates['longitude'].values))
print( longitude_extents )
```

```
(118.15356777397116, 122.86595009140785)
```

In [7]:
```
time_extents = (min(coordinates['time'].values),max(coordinates['time'].values))
print( time_extents )
```

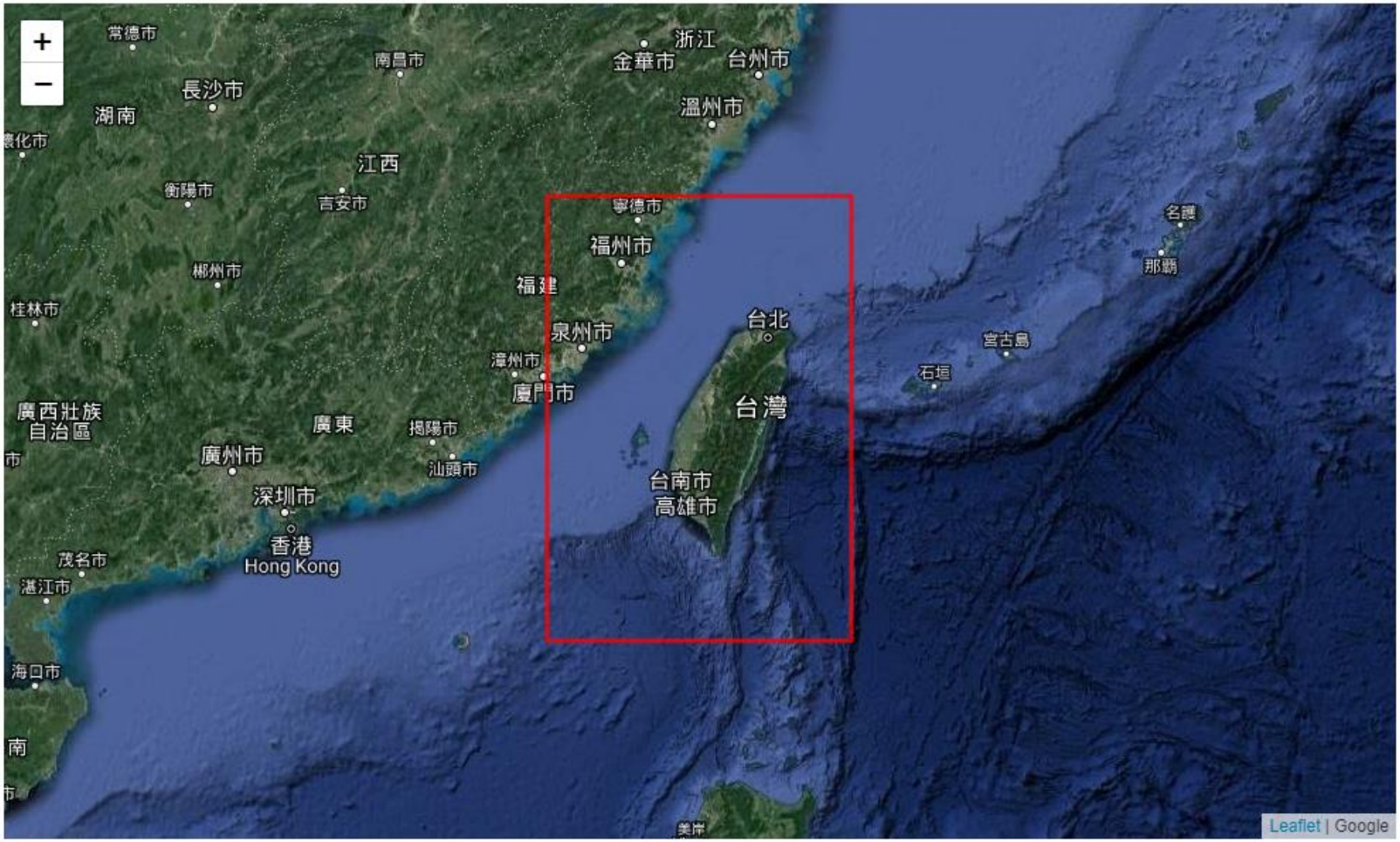
```
(numpy.datetime64('2000-02-07T02:19:20.825028000'), numpy.datetime64('2019-03-24T02:14:16.044100000'))
```

Python DataCube

Visualize Data Cube Region

In [8]:
```
## The code below renders a map that can be used to orient yourself with the region.
from utils.data_cube_utilities.dc_display_map import display_map
display_map(latitude = latitude_extents, longitude = longitude_extents)
```

Out[8]:

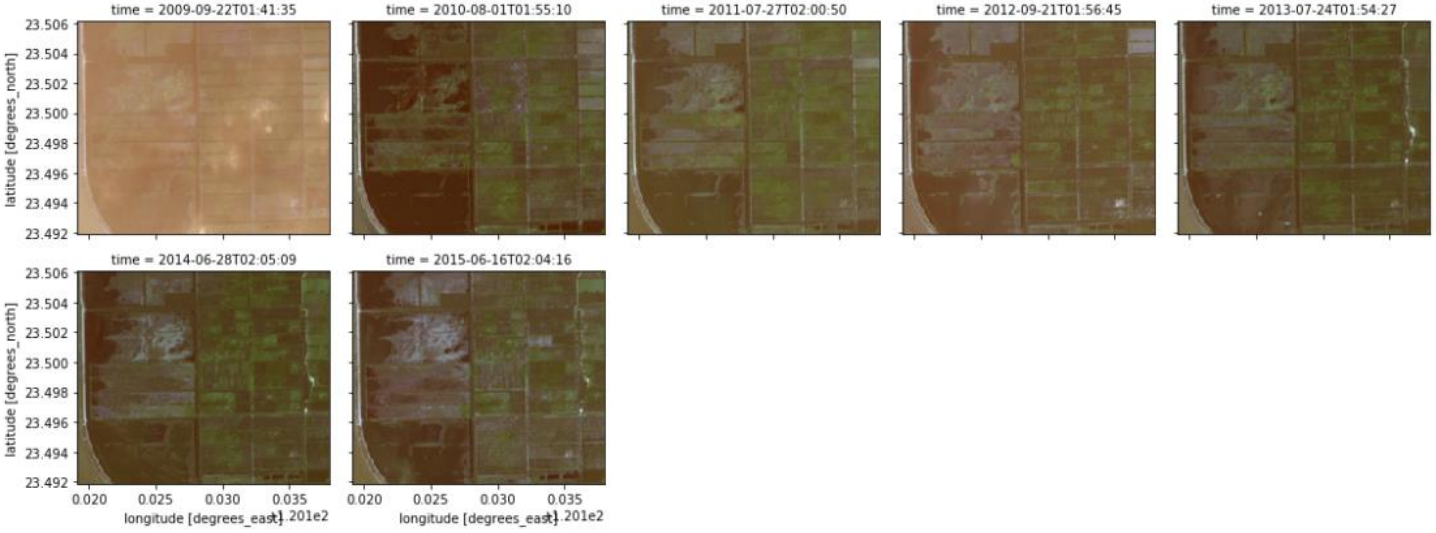


Python DataCube

Drawing RGB color images

In [18]:
```
from datacube.storage.masking import mask_invalid_data

query = {
    'time': time_extents,
    'lat': latitude_extents,
    'lon': longitude_extents,
}

time_extents = ('2009-09-06', '2016-06-19')

data = dc.load(product=product,platform = platform, measurements=['blue', 'green', 'red'], **query)
data = mask_invalid_data(data)

fake_saturation = 4000
rgb = data.to_array(dim='color')
rgb = rgb.transpose(*(rgb.dims[1:]+rgb.dims[:1]))  # make 'color' the last dimension
rgb = rgb.where((rgb <= fake_saturation).all(dim='color'))  # mask out pixels where any band is 'saturated'
rgb /= fake_saturation  # scale to [0, 1] range for imshow

rgb.plot.imshow(x=data.crs.dimensions[1], y=data.crs.dimensions[0],
                col='time', col_wrap=5, add_colorbar=False)
```

Out[18]: <xarray.plot.facetgrid.FacetGrid at 0x7fcd4c32a630>



Python DataCube

Export Data

Export to GeoTIFF

In [14]:

```
from utils.data_cube_utilities.import_export import export_xarray_to_geotiff

# Ensure the output directory exists before writing to it.
if platform == 'FS2_MS_NSPO':
    !mkdir -p ./output/geotiffs/fs2
else:
    !mkdir -p ./output/geotiffs/fs2

output_path = "./output/geotiffs/fs2/fs2"

export_xarray_to_geotiff(combined_dataset, output_path)
```

jupyter

Quit Logout

Files Running Clusters

Duplicate Rename Move Download View Edit

Upload New

1 / twdc_Training / output / geotiffs / fs2

Name	Last Modified	File size
..	幾秒前	
fs2_2009_09_22_01_41_35.tif	6 天前	1.23 MB

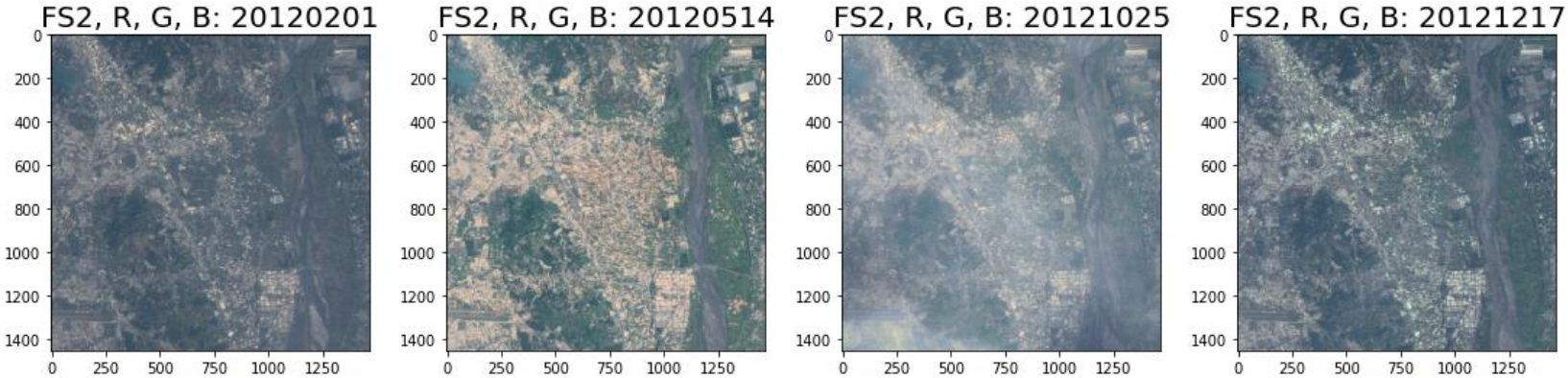
Python DataCube

■ Cloud filter

```
measurements = ['red', 'green', 'blue', 'nir']
nTotal=0

plt.figure(figsize=[9.0,9.0])
for p in lstOK:
    dataset=fs2_dataset.isel(time=p)
    ds=dataset['pixel_qa'].values
    cmask=((ds==2))*1
    cmask3=np.repeat(cmask[:, :, np.newaxis], 3, axis=2)
    m,n=cmask.shape
    rPercent=np.sum(cmask)/m/n*100



    day='%4d%02d%02d' % (lstYEAR[p], lstMONTH[p], lstDAY[p])
    center_dt='%4d-%02d-%02d %02d:%02d:%02d.00' % (lstYEAR[p], lstMONTH[p], lstDAY[p], lstHOUR[p], lstMINUTE[p], lstSECOND[p])
    from_dt=  '%4d-%02d-%02d %02d:%02d:%02d.00' % (lstYEAR[p], lstMONTH[p], lstDAY[p], lstHOUR[p], lstMINUTE[p], lstSECOND[p]-1)
    to_dt=    '%4d-%02d-%02d %02d:%02d:%02d.00' % (lstYEAR[p], lstMONTH[p], lstDAY[p], lstHOUR[p], lstMINUTE[p], lstSECOND[p]+1)


    if rPercent>=80:

        nTotal=nTotal+1

        ######## geotif 檔案位置設定
        path="02_FS_geotif_download/FS2_"+day+"/"
        subprocess.call(['mkdir',path])
        write_geotiff_from_xr(path+"S2_"+day+".tif", dataset, measurements)

        fig=plt.figure()
```

Python DataCube

- Before cloud filter script: 31 files in total
- After cloud filter script: 4 files in total merely



Reference

- OpenDataCube official website

https://www.opendatacube.org/

- Installing the Open Data Cube

https://www.opendatacube.org/installation

- Open Data Cube Manual

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

- API reference

https://opendatacube.readthedocs.io/en/latest/dev/api/index.html

- Video

https://www.youtube.com/watch?v=DZ40HNq47ro

Q&A