

Finding, downloading and decoding data as a WIS2 Data Consumer



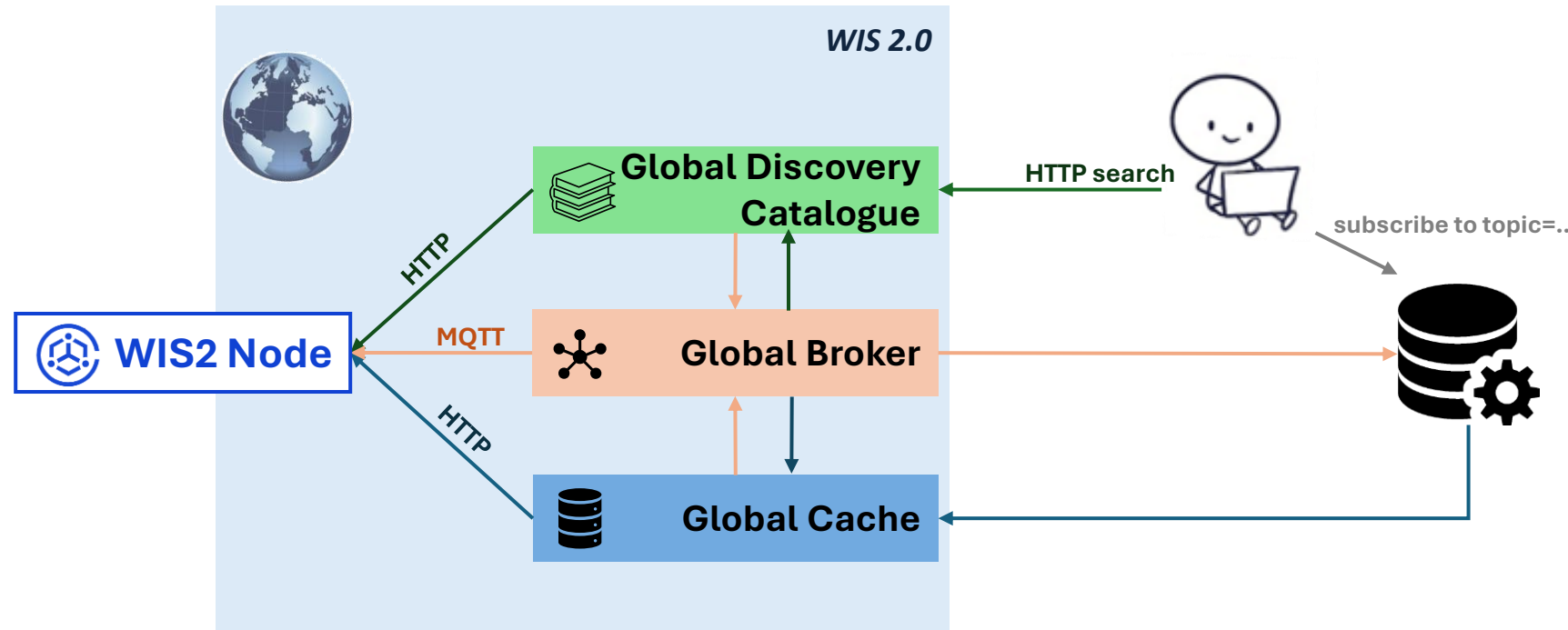
WORLD
METEOROLOGICAL
ORGANIZATION



Finding data on WIS2

The Global Discovery Catalogue (GDC) provides cataloguing and discovery capability for WIS2 datasets

- Web-based API facilitating search/browse data published via WIS 2.0 (OGC API - Records)
- Harvests WIS2 discovery metadata from WIS2 Nodes (WCMP2 / OGC API - Records)
- Yellow pages (discovery metadata) gateway into WIS2 data and services
- Provides indexing capability to mass market search engines
- Provides quality assessment services of discovery metadata in support of continuous improvement in alignment with WIS2 metadata Key Performance Indicators (KPIs)



Access data using OGC API

Each Global Discovery Catalogue provides an OGC API endpoint to query records

GDC CMA, China:

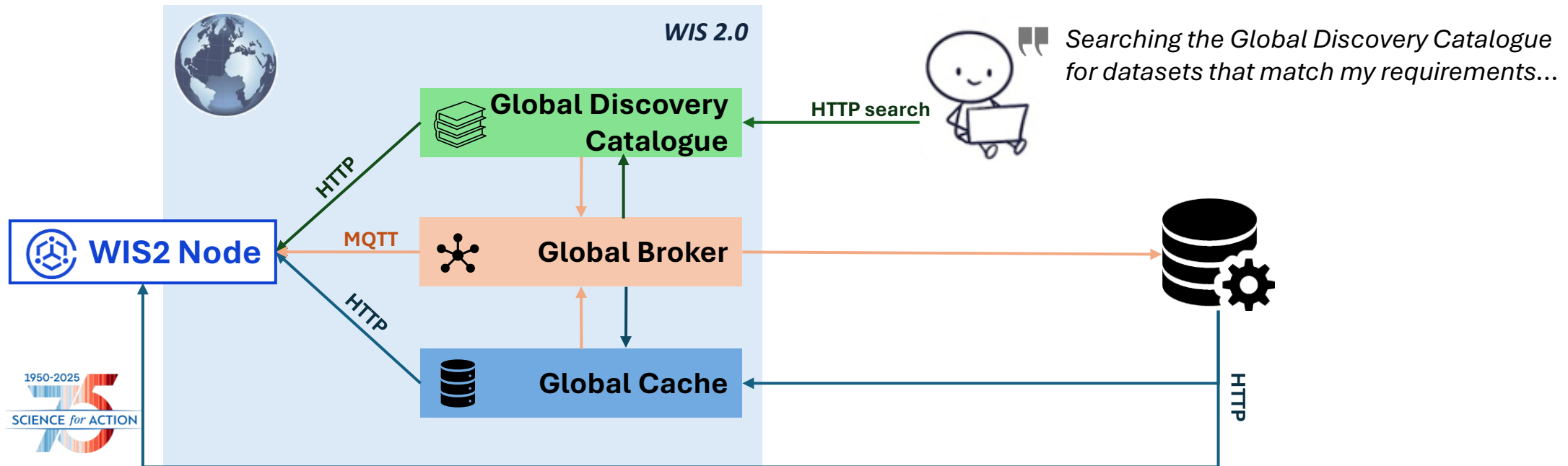
<https://gdc.wis.cma.cn/collections/wis2-discovery-metadata/items>

GDC DWD, Germany:

<https://wis2.dwd.de/gdc/collections/wis2-discovery-metadata/items>

GDC ECCC, Canada:

<https://wis2-gdc.weather.gc.ca/collections/wis2-discovery-metadata/items>



Searchable Catalogue

Predicate	Example(s)
Spatial	<code>bbox=minx,miny,maxx,maxy</code>
Temporal	• <code>datetime=t1</code> • <code>datetime=t1/t2</code> • <code>datetime=t1/..</code>
Freetext	<code>q=air+temperature</code>
Attributes	• <code>title=air+temperature</code> • <code>contacts.address.country=Oman</code>
Sorting/paging	• <code>sortby=title&limit=100&startindex=11</code> • <code>"prev"</code> / <code>"next"</code> link relations
Filter returnable properties	• <code>properties=title,description</code>

pywiscat

pywiscat provides a Pythonic API atop the WMO WIS2 Catalogue in support of reporting and analysis of the WIS2 Catalogue and its associated discovery metadata

github.com/wmo-im/pywiscat

```
# fetch version
pywiscat --version
```

```
## WIS2 workflows
```

```
# search the WIS2 Global Discovery Catalogue (GDC)
pywiscat search
```

```
# search the WIS2 Global Discovery Catalogue (GDC) with a full text query
pywiscat search --query radar
```

```
# search the WIS2 Global Discovery Catalogue (GDC) for only recommended data
pywiscat search --data-policy recommended
```

```
# search the WIS2 Global Discovery Catalogue (GDC) with a bounding box query
pywiscat search --bbox -142,42,-52,84
```

```
# get more information about a WIS2 GDC record
pywiscat get urn:x-wmo:md:can:eccc-msc:c7c9d726-c48a-49e3-98ab-78a1ab87cda8
```

```
## WIS2 workflows
```

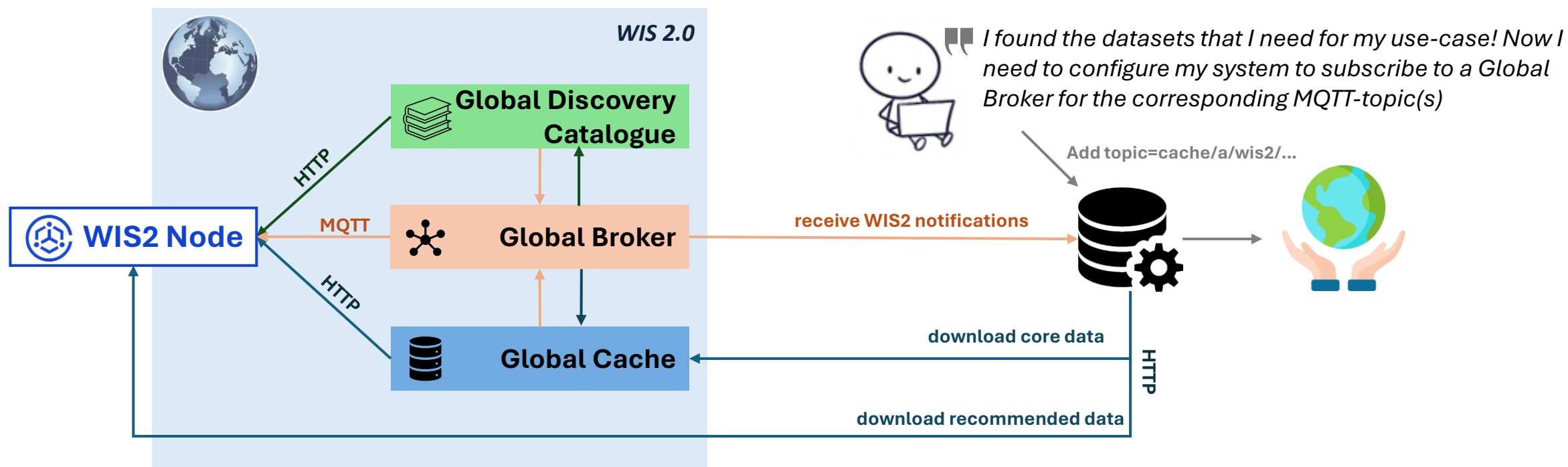
```
from pywiscat.wis2.catalogue import search, get
```

```
# search catalogue
results = search(q='radar', bbox=[-142, 42, -52, 84]))
```

```
# get a single catalogue record
results = get('urn:x-wmo:md:can:eccc-msc:c7c9d726-c48a-49e3-98ab-78a1ab87cda8')
```

Processing WIS2 notifications as a Data Consumer

Data Consumers use MQTT client-software to subscribe to specific channels and process data in real-time



Data access

WIS2 Notifications provide a canonical link to access data

The canonical link is a “one click” link to download data to your laptop/desktop/system

When you subscribe to data notifications, use the canonical link to download the data:



```
{  
  "href": "https://example.org/data-file.bufr",  
  "rel": "canonical",  
  "type": "application/bufr"  
}
```

Data publisher can indicate that previously published data has been updated and should be overwritten by using rel=update

```
{  
  "href": "https://example.org/data-file.bufr",  
  "rel": "update",  
  "type": "application/bufr"  
}
```

Custom MQTT client scripting



Example Python code using paho-mqtt

```
import json
import paho.mqtt.client as mqtt_client

def on_message(client, userdata, message):
    j = json.load(message.payload.decode('utf-8'))
    for link in j['links']:
        if link['rel'] == 'canonical':
            client_specific_function(link['href'])

client = mqtt_client.Client('MyDataClientID')
client.username_pw_set('everyone', 'everyone')
client.tls_set(tls_version=2)
client.connect('globalbroker.meteo.fr', port=8883)
client.on_message = on_message
client.subscribe('origin/a/wis2/int-ecmwf/data/core/weather/prediction/forecast/cyclone_tracks')
```

Data Consumer can customize their "on_message" code to define the actions specific for their data ingestion

Options to subscribe & download on WIS2



CUSTOM DEVELOPMENT:

Data Consumers may prefer to create a custom MQTT-client based workflow to:

- **Integrate** WIS2 subscriber within **existing data processing system**
- **Parse and process data directly**, without writing to file-system
- **Apply authentication** for restricted datasets (data-policy=recommended)

MQTT clients are available for different software environments

One widely used client for Python is paho-mqtt

EXISTING OPEN-SOURCE SOLUTIONS:

wis2downloader: github.com/wmo-im/wis2downloader

pywis-pubsub: github.com/wmo-im/pywis-pubsub

wis2downloader



wis2downloader:

- Flask-based Python application including MQTT-client
- API to manage subscriptions on MQTT-topics
- Multiple download workers for more efficient data downloading
- Provides statistics via /metrics endpoint for Prometheus



Flask

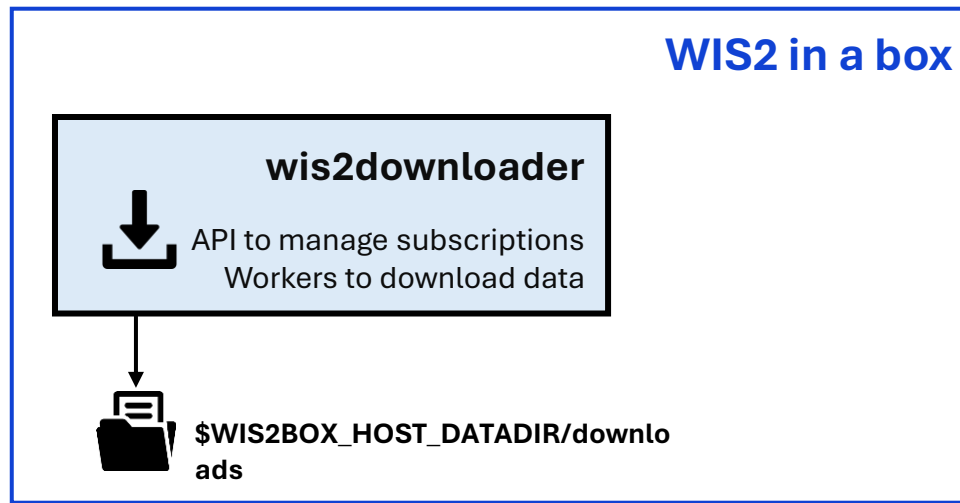


Included in wis2box, can also be installed independently: `pip3 install wis2downloader`

wis2downloader in wis2box

wis2downloader included within wis2box-stack as a separate docker container

- data is downloaded in \$WIS2BOX_HOST_DATADIR/downloads
- additional wis2downloader configuration using wis2box.env



Login to wis2downloader to manage subscriptions using CLI

```
python3 wis2box-ctl.py login wis2downloader
```

Commands available in wis2downloader-container:

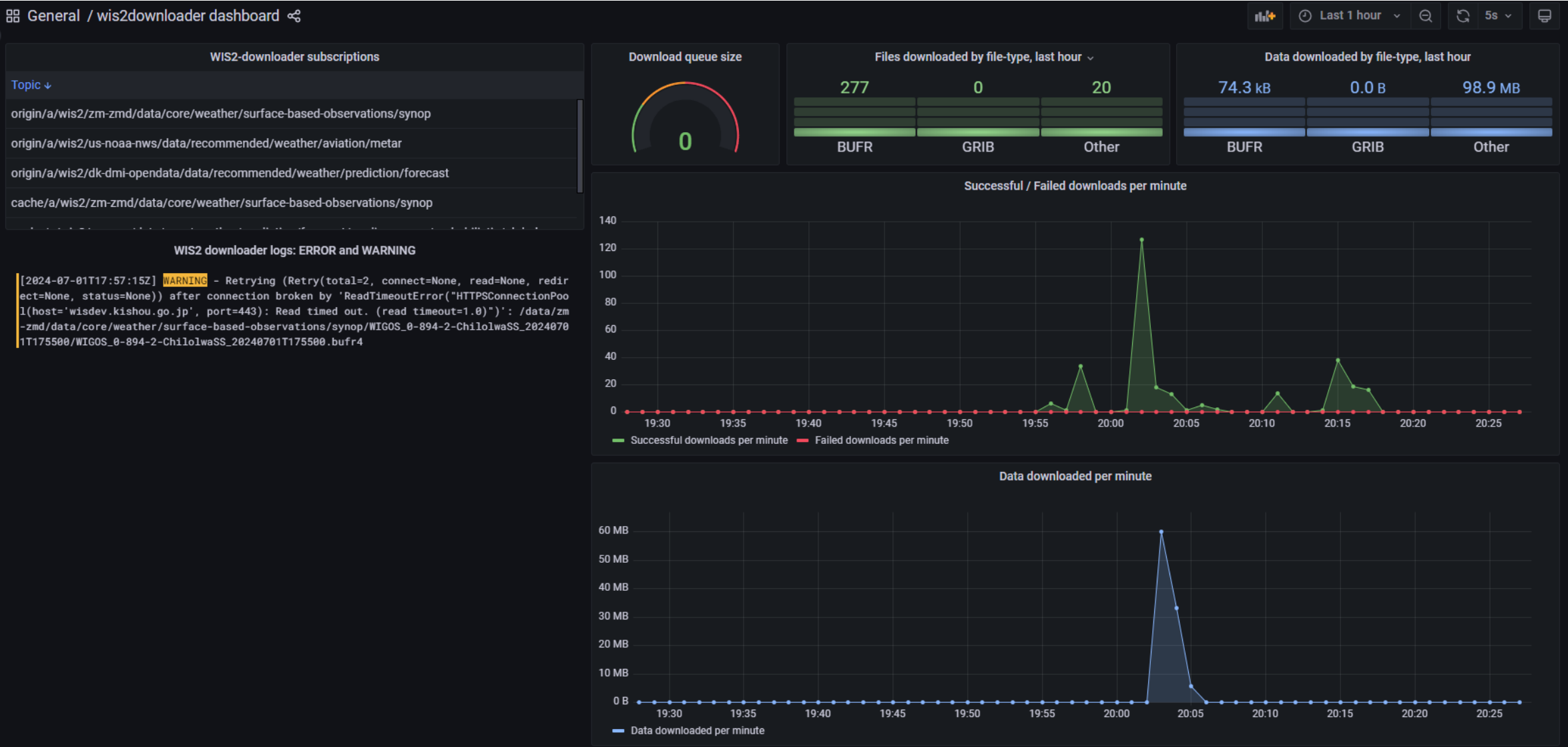
```
wis2downloader list-subscriptions
```

```
wis2downloader add-subscription --topic <topic>
```

```
wis2downloader remove-subscription --topic <topic>
```

wis2downloader in wis2box

The monitoring stack in wis2box collects and stores wis2downloader metrics in Prometheus and provides a dashboard for viewing information about the data downloaded:



Consuming data downloaded from WIS2: data format and content

Data shared on WIS2 should be formatted with WMO data standards, e.g. BUFR4, GRIB2, NetCDF, WaterML, ...

WMO data standards are described in the [*Manual on Codes*](#) (WMO-No. 306), Volume I.2 and the [*Manual on Codes*](#) (WMO-No. 306), Volume I.3.

Requirements for data standards are governed by different WMO programs, such as WIGOS for observations and WIPPS for prediction-based products.

Descriptions about these formats and the contents of the dataset should be available in the discovery metadata.

Manual on Codes (WMO-No. 306), Volume I.2 - Part B and Part C

Annex II to the WMO Technical Regulations

Coded messages are used for the international exchange of meteorological information comprising observational data provided by the WMO Integrated Global Observing System and processed data provided by the WMO Integrated Processing and Prediction System (formerly Global Data-processing and Forecasting System). Coded messages are also used for the international exchange of observed and processed data required in specific applications of meteorology to various human activities and for exchanges of information related to meteorology.



Manual on Codes (WMO-No. 306), Volume I.3 - Part D

Annex II to the WMO Technical Regulations

Representations derived from data models consist of the specification of the list of standard representations derived from data models, including those using extensible markup language (XML), with their specifications and associated code tables.



Tools for decoding WMO data formats

Libraries for decoding BUFR & GRIB2:

ecCodes - ECMWF's official library for decoding/encoding GRIB1/2 and BUFR3/4 (**C, Fortran, Python**)

NCEPLIBS-bufr - NOAA's library for encoding/decoding BUFR messages (**Fortran, C**).

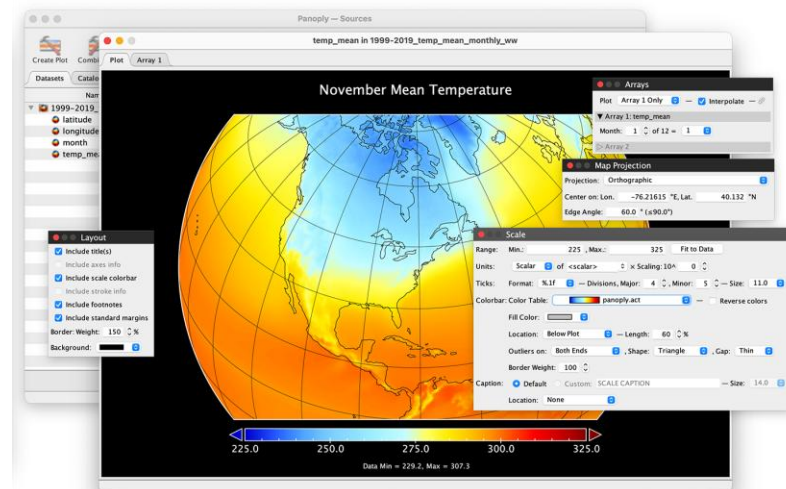
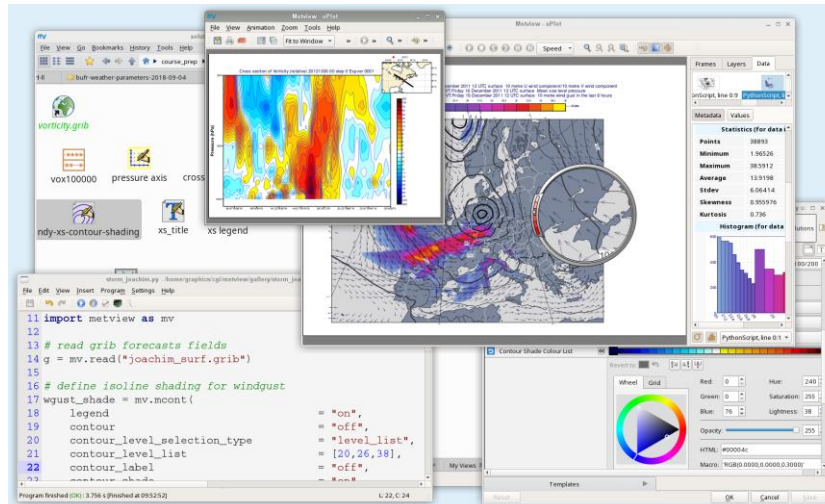
wgrib2 - NOAA's Utility and C library for decoding, subsetting, and regridding GRIB2 data (**C**)

No-code solutions:

Panoply - a cross-platform application that plots geo-referenced and other arrays from NetCDF, HDF, GRIB, and other datasets

Metview - ECMWF's application for data analysis and visualization, which supports GRIB and BUFR formats

Integrated Data Viewer (IDV) - a free Java-based software framework for analyzing and visualizing geoscience data, including support for GRIB and NetCDF formats



Summary: finding, download and decoding data on WIS2

Find new data by searching one of the Global Discovery Catalogue

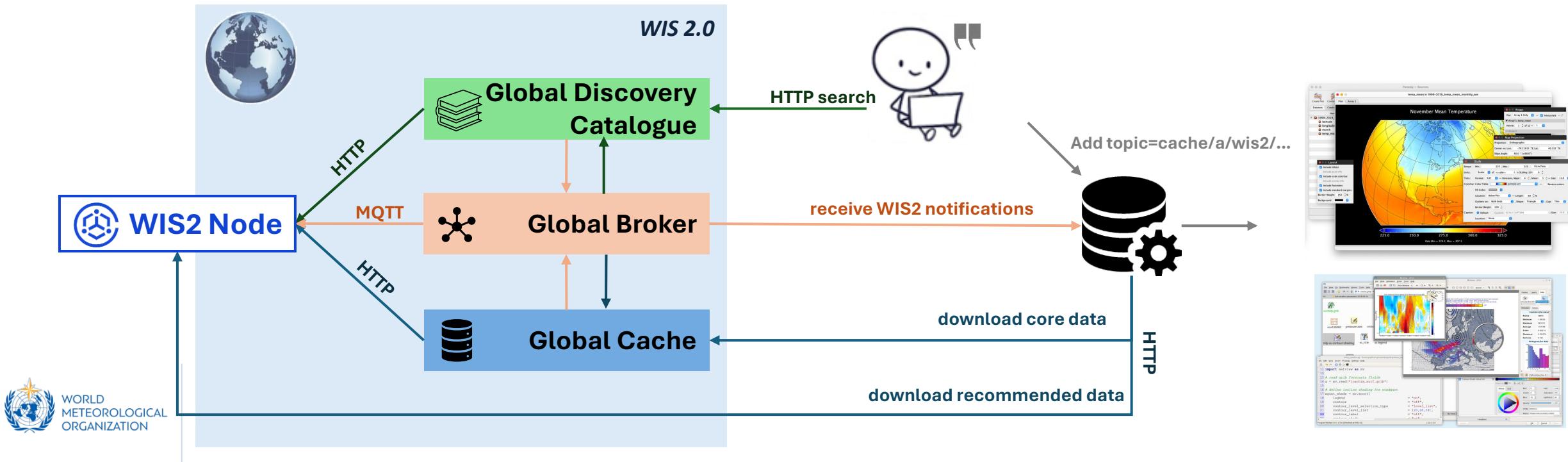
- Query the OGC API endpoint over HTTP or use tools like 'pywiscat'

Setup an MQTT subscription and use the URL in the WIS2 notification to download the data

- Develop custom code using MQTT client libraries or use tools like 'wis2downloader'

Process and decode the received data using different tools depending on the format

- Various software available for decoding WMO binary data formats



Thank you



WORLD
METEOROLOGICAL
ORGANIZATION

