WIS2 Cookbook

World Meteorological Organization

Date: 2025-10-18

Version: 0.1.0

Document status: DRAFT

Document location: https://wmo-im.github.io/wis2-cookbook/cookbook/wis2-cookbook-DRAFT.html

Standing Committee on Information Management and Technology (SC-IMT)[1]

Commission for Observation, Infrastructure and Information Systems (INFCOM)[2]

Table of Contents

Chapter 1. Introduction

1.1. Purpose

In conjunction with the Manual on the WMO Information System volume II (WMO-No. 1060) (Manual on WIS volume II: WIS 2.0), the present Guide to the WMO Information System volume II (Guide to WIS volume II: WIS 2.0) is designed to ensure adequate uniformity and standardization in the data, information and communication practices, procedures and specifications employed by Members of the World Meteorological Organization (WMO) in the operation of the WMO Information System WIS 2.0 as it supports the mission of the Organization. The Manual on WIS contains standard and recommended practices, procedures and specifications. The Guide to WIS contains additional information concerning practices, procedures, and specifications that Members are invited to follow or implement in establishing and conducting their arrangements in compliance with the WMO Technical Regulations and in developing meteorological and hydrological services.

This cookbook provides various code snippets, recipes and workflow examples in support of WIS2 requirements.

1.2. Key documents

WIS2 Manuals and Guides can be found at the following locations:

- Manual on the WMO Information System
- Guide to the WMO Information System Volume II - WMO Information System 2.0
- Provisions for the Transition from the WMO Information System (WIS) 1.0 and Global Telecommunication System to WIS 2.0

1.3. Publication and contribution

Updates to this cookbook will be published twice per year - in March and October.

All "recipes" published in this cookbook have been reviewed by experts from the INFCOM Standing Committee on Information Management and Technology (SC-IMT). Publication of a new release is approved by the Chair of SC-IMT.

NOTE	The WIS2 Cookbook is not regulatory material and does not follow the approval and publication process for the WIS2 Manual and Guide.

The WIS2 community are invited to provide feedback on the "recipes" in this Cookbook and also to propose new recipes. Contributions should be made by raising issues in the wmo-im/wis2-cookbook GitHub repository. Experts from SC-IMT will review suggestions with the aim to incorporate them into the next release of the WIS2 Cookbook.

[1] https://community.wmo.int/governance/commission-membership/commission-observation-infrastructures-and-information-systems-infcom/commission-infrastructure-officers/infcom-management-group/standing-committee-information-management-and-technology-sc-imt

[2] https://community.wmo.int/governance/commission-membership/infcom

Chapter 2. Recipes for data consumers

2.1. Searching the Global Discovery Catalogue

The Global Discovery Catalogue (GDC) allows for a wide range of query predicates to search for data in WIS2 as per the OGC API - Records - Part 1: Core specification.

The GDC can be searched via the `/collections/wis2-discovery-metadata/items` endpoint. This endpoint provides a number query parameters as described in the examples below.

NOTE: examples below are not URL encoded for clarity / readability, but should be when interacting with the GDC.

2.1.1. Spatial queries

- search for metadata records of data in Canada: `bbox=-142,42,-52,84`

Note that the format of `bbox` is comma-separated values in the following order:

- minimum longitude
- minimum latitude
- maximum longitude
- maximum latitude

2.1.2. Temporal queries

- search for metadata records updated since 29 July 2024: `datetime=2024-07-29/..`
- search for metadata records updated before 29 July 2024: `datetime=../2024-07-29`
- search for metadata records updated on 29 July 2024: `datetime=2024-07-29`

2.1.3. Equality queries

- search for metadata records whose title contains the terms hourly observations: `title=hourly observations`
- search for metadata records whose title contains the terms hourly or observations: `title=hourly | observations`
- search for metadata records for a specific contact organization `contacts.organization=Direction Generale de la Météorologie`

2.1.4. Freetext search

- search metadata records for temperature: `q=temperature`
- search metadata records for GRIB2 data: `q=GRIB2`
- search metadata records for any GRIB data: `q=*GRIB*`

- search metadata records for any GRIB data in Germany: `q=*GRIB* AND germany`
- search for either GRIB data or data in Europe with subscriptions to the Météo-France Global Broker: `q=(*GRIB* OR *Europe*) AND *globalbroker*`
- search for data from Belize with MQTT subscription capabilities: `q="cache/a/wis2/bz-nms"`

2.1.5. Sorting

- sort search results by title, ascending: `sortby=title`
- sort search results by title, descending: `sortby=-title`

2.1.6. Paging

- present search results 1-10: `limit=10`
- present search results 11-20: `limit=10&offset=10`
- limit to 3 search results: `limit=3`

2.2. Finding data subscription services from the Global Discovery Catalogue

The Global Discovery Catalogue (GDC) contains both real-time and non real-time data. The WMO Core Metadata Profile (WCMP2) allows for description of real-time data via its distribution information, which data publishers use to describe and define connectivity and subscription information for a given dataset.

A typical WCMP2 distribution link for data subscriptions can be found below:

```
{
    "rel": "items",
    "href": "mqtts://everyone:everyone@globalbroker.meteo.fr:8883",
    "channel": "origin/a/wis2/ca-eccc-msc/data/core/hydrology",
    "type": "application/geo+json",
    "title": "Data notifications"
}
```

NOTE The `channel` property represents WIS2 topic which can be used to subscribe to the `href` property (i.e. the MQTT address) of the Global Broker (GB).

Programmatically, a GDC client can query the catalogue and filter the results for real-time subscriptions in the following manner:

```
import requests

response = requests.get('https://wis2-gdc.weather.gc.ca/collections/wis2-discovery-
metadata/items').json()
```

```
def is_wis2_subscription_link(link) -> bool:
    if (link['href'].startswith('mqtt') and
            link.get('channel', '').startswith(('origin/a/wis2', 'cache/a/wis2'))):
        return True


for feature in response['features']:
    for link in feature['links']:
        if is_wis2_subscription_link(link):
            print('WIS2 subscription link')
```

Using the `href` and `channel` properties of a matching link object, a client can connect and subscribe to data notifications for a given dataset.

Chapter 3. Recipes for data publishers

3.1. Validating a WIS2 Notification Message

A WIS2 Notification Message provides a JSON Schema which can be used by any programming language that supports JSON and JSON Schema validation.

Using Python and check-jsonschema

```
# install check-jsonschema Python Package from the Python Package Index (PyPI)
pip3 install check-jsonschema

# download WNM schema
curl -O http://schemas.wmo.int/wnm/1.1.0/schemas/wis2-notification-message-
bundled.json

# run schema validation
check-jsonschema --schemafile wis2-notification-message-bundled.json
/path/to/my/wnm.json
```

The pywis-pubsub tool provides a test suite to validate a message against the WNM specification requirements, as well as a Python API for application integration. Consult the pywis-pubsub README on GitHub for more information/examples.

Using pywis-pubsub

```
# install pywis-pubsub
pip3 install pywis-pubsub

# sync WIS2 notification schema
pywis-pubsub schema sync

# validate WNM against abstract test suite (file on disk)
pywis-pubsub ets validate /path/to/file.json

# validate WNM against abstract test suite (URL)
pywis-pubsub ets validate https://example.org/path/to/file.json
```

3.2. Publishing a WIS2 Notification Message with access control

Recommended data in WIS2 may be open or access controlled. For data publication with access control implications, WNM provides a `security` object as part a link object. The `security` object is defined using OpenAPI Security Scheme definitions.

Access control using HTTP Basic authentication

```
{
  "rel": "canonical",
  "type": "application/grib2",
  "href": "https://example.org/my/protected/data/nwp/12/003/20240805120000-air-temp-
500.grib2",
  "security": {
    "default": {
      "type": "http",
      "scheme": "basic",
      "description": "Please contact us for access information"
    }
  }
}
```

Access control using an API key

```
{
  "rel": "canonical",
  "type": "application/geo+json",
  "href": "https://example.org/my/protected/data/nwp/12/003/20240805120000-air-temp-
500.grib2",
  "security": {
    "default": {
      "type": "apiKey",
      "name": "api-key",
      "in": "query",
      "description": "Please see https://example.org/contact-us for more information"
    }
  }
}
```

Note:

- the child property under `security` (`default` in the examples above) can be any text or label. We use `default` here as a convention
- the `security.default.name` is the name of the API key parameter as defined by your API service
- only properties defined in the OpenAPI Security Scheme definition are allowed. Any additional properties will invalidate the WNM

Of course, always ensure your WNM is valid (see [Validate a WIS2 Notification Message] for more information).

3.3. Publishing a WIS2 Notification Message with embedded data

A WIS2 Notification Message provides the ability to embed data as part of a notification (see the Properties / Content section of the specification).

Providing embedded data offers users the ability to access the data without the need to execute the additional step of downloading the data via the actionable link (see the Links section of the specification).

In some cases, a data publisher also prefer not to publish the corresponding actionable link to infrastructure (e.g. an HTTP or FTP server).

When the data publisher decides to not publish an actionable link and to rely exclusively on the embedded data feature, the data publisher will:

- use `properties.content` object to provide the data
- use `properties.cache=false` to instruct WIS2 Global Cache Services to not cache the data (note: `properties.cache` is not required for recommended data, since recommended data is not cached by WIS2 Global Caches)
- provide a link object whose `href` results in HTTP 204 (No Content)

To provide an HTTP 204 link, there a couple of options:

Option 1: configure your web server to provide an endpoint that always returns 204:

Nginx example configuration

```
# always return HTTP 204 when accessing /no-content-here
location /no-content-here/ {
  return 204;
}
```

Apache example configuration

```
# always return HTTP 204 when accessing /no-content-here
Redirect 204 no-content-here
```

which then would result in the link object as follows:

```
{
    "href": "https://example.org/no-content-here",
    "rel": "canonical"
}
```

Option 2: use a service that provides HTTP responses as a service. One such service is the HTTP Response API, which then would result in the link object as follows:

```
{
    "href": "https://http.codes/204",
    "rel": "canonical"
}
```

Embedded data example:

```
{
    "id": "31e9d66a-cd83-4174-9429-b932f1abe1be",
    "conformsTo": [
        "http://wis.wmo.int/spec/wnm/1/conf/core"
    ],
    "type": "Feature",
    "geometry": {
        "type": "Point",
        "coordinates": [
            6.146255135536194,
            46.223296618227444
        ]
    },
    "properties": {
        "pubtime": "2022-03-20T04:50:18Z",
        "datetime": "2022-03-20T04:45:00Z",
        "integrity": {
            "method": "sha512",
            "value": "A2KNxvks...S8qfSCw=="
        },
        "cache": false,
        "data_id": "dataset/123/data-granule/UANT01_CWAO_200445___15103.bufr4",
        "metadata_id": "urn:wmo:md:ca-eccc-msc:observations.swob",
        "content": {
            "encoding": "utf-8",
            "value": "encoded bytes from the file",
            "size": 457
        }
    },
    "links": [
        {
            "href": "https://http.codes/204",
            "rel": "canonical"
        }
    ]
}
```

For cases of publishing recommended data, the same approaches/example can be used, without the need to set `properties.cache`.

3.4. Publishing a WIS2 Notification Message for resource deletion

A WIS2 Notification Message provides the ability to publish notifications for new, updated or deleted data and metadata (see the Links section of the specification).

Similar to the embedded data Recipe, for data or metadata deletions (core or recommended data), a data publisher may not wish to manage or publish the data or metadata link to infrastructure to an HTTP or FTP server.

In this case, a similar strategy can be used to provide an HTTP 204 No Content link, as per the embedded data Recipe, along with setting `rel=deletion` in the link object.

Resource deletion example:


```
{
    "id": "31e9d66a-cd83-4174-9429-b932f1abe1be",
    "conformsTo": [
        "http://wis.wmo.int/spec/wnm/1/conf/core"
    ],
    "type": "Feature",
    "geometry": {
        "type": "Point",
        "coordinates": [
            6.146255135536194,
            46.223296618227444
        ]
    },
    "properties": {
        "pubtime": "2022-03-20T04:50:18Z",
        "datetime": "2022-03-20T04:45:00Z",
        "integrity": {
            "method": "sha512",
            "value": "A2KNxvks...S8qfSCw=="
        },
        "cache": false,
        "data_id": "dataset/123/data-granule/UANT01_CWAO_200445___15103.bufr4",
        "metadata_id": "urn:wmo:md:ca-eccc-msc:observations.swob"
    },
    "links": [
        {
            "href": "https://http.codes/204",
            "rel": "deletion"
        }
    ]
}
```

3.5. Publishing a WMO Core Metadata Profile record for retired data

The primary target for WCMP2 records is for real-time or archive data discovery, access and visualization, with the understanding that a dataset (and its associated discovery metadata) is properly managed and kept up to date. In certain cases, datasets may become unavailable; this could mean that the dataset lifecycle has reached end of life, and/or no longer maintained.

An "end of life" or retired dataset should be independent from its description. That is, even if a dataset is retired, there is value in having discovery metadata for these datasets, and so the WCMP2 record associated with it should not be deleted from the WIS2 Global Discovery Catalogues. The FAIR Data Principles (A2) also state that metradata should persist even the data are no longer sustained [1].

To describe a retired dataset in WCMP2, one can use the `properties.status` object as per below:

Describing a retired dataset in WCMP2

```
"properties": {
  "status": {
    "id": "retired",
    "title": "Retired",
    "url": "https://standards.iso.org/iso/19115/-
3/mcc/1.0/codelists.xml#MD_ProgressCode_retired"
  }
}
```

The above status is defined by ISO 19115-3 as "*item is no longer recommended for use. It has not been superseded by another item*" [2].

Another aspect to consider reviewing all links in the WCMP2 record (`links` object). For example, MQTT links can be removed, and any links which no longer provide the data can also be removed. If the data is retired and has alternative access mechanisms (email, web form), these links can be added to the record.

3.6. Validating a WMO Core Metadata Profile record

The pywcmp tool provides a test suite to validate a message against the WCMP2 specification requirements, as well as a Python API for application integration. Consult the pywcmp README on GitHub for more information/examples.

Using pywcmp for WCMP2 validation

```
# install pywcmp
pip3 install pywcmp

# sync WCMP2 schemas and codelists
pywcmp bundle sync
```

```
# validate WCMP2 against abstract test suite (file on disk)
pywcmp ets validate /path/to/file.json

# validate WCMP2 against abstract test suite (URL)
pywcmp ets validate https://example.org/path/to/file.json
```

A WCMP2 record can also be validated using pywcmp "as a service" using the Canadian and German WIS2 Global Discovery Catalogues, which provide an online validator:

- Navigate to https://wis2-gdc.weather.gc.ca/openapi?f=html or https://wis2.dwd.de/gdc/openapi?f=html
- Navigate to section **pywcmp-wis2-wcmp2-ets**, endpoint `/processes/pywcmp-wis2-wcmp2-ets/execution` (POST)
- Click "Try it out"
- In the section "Mandatory execute request JSON", paste the WCMP2 JSON inside the `record` object

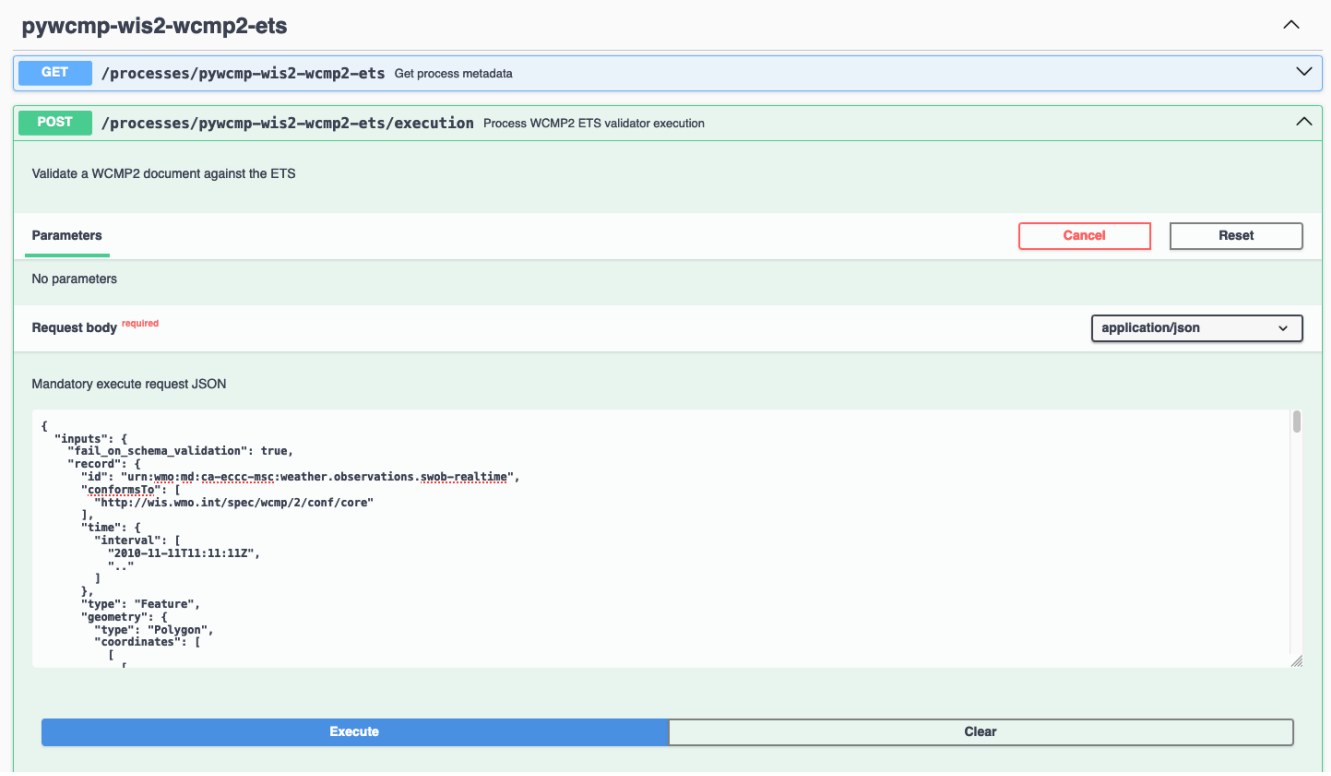

Figure 1. WIS2 GDC online validator, request via WCMP2 copy/paste

Alternatively, the validator service also accepts a URL (if the WCMP2 record is online):

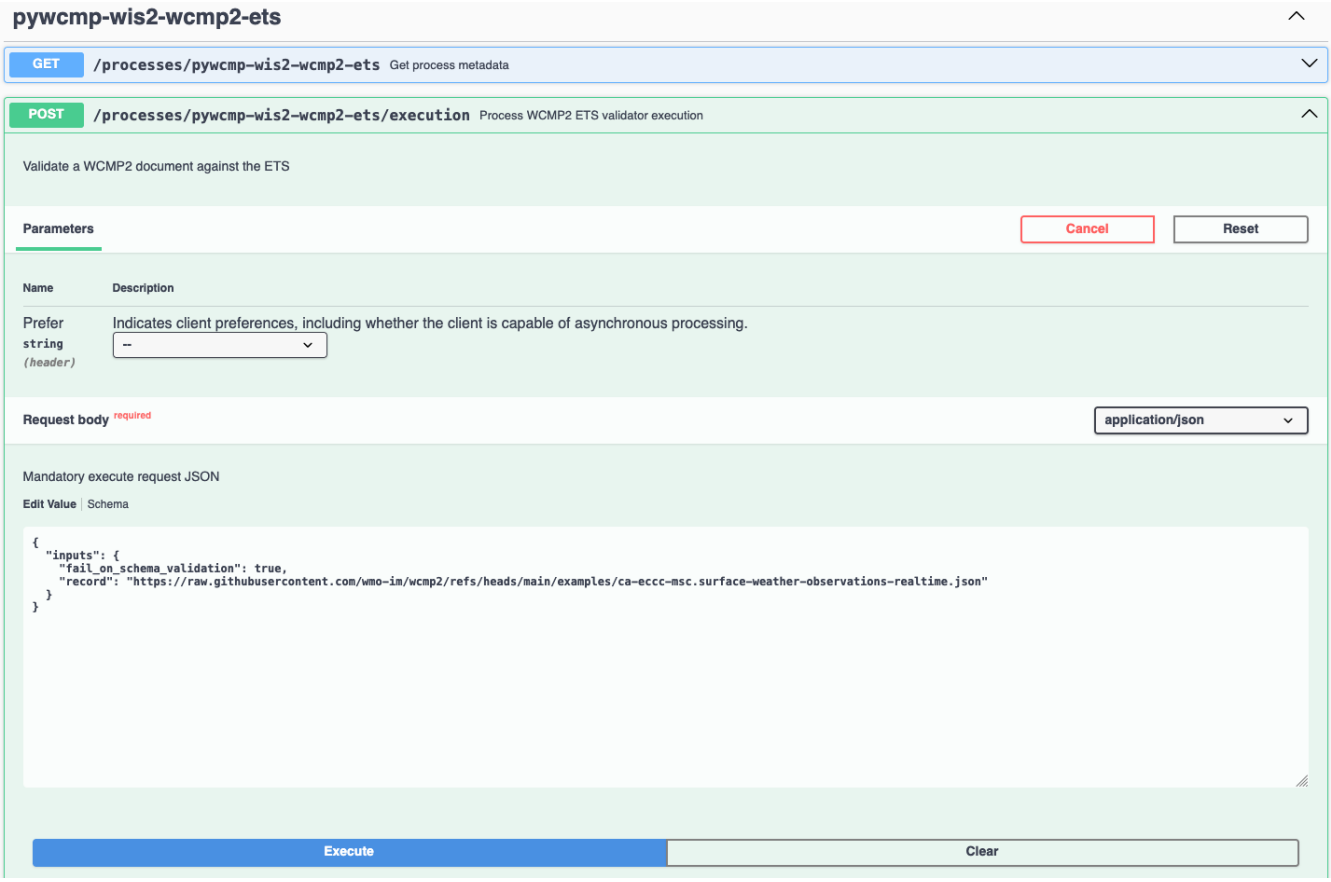

Figure 2. WIS2 GDC online validator, request via WCMP2 URL

- Click "Execute"

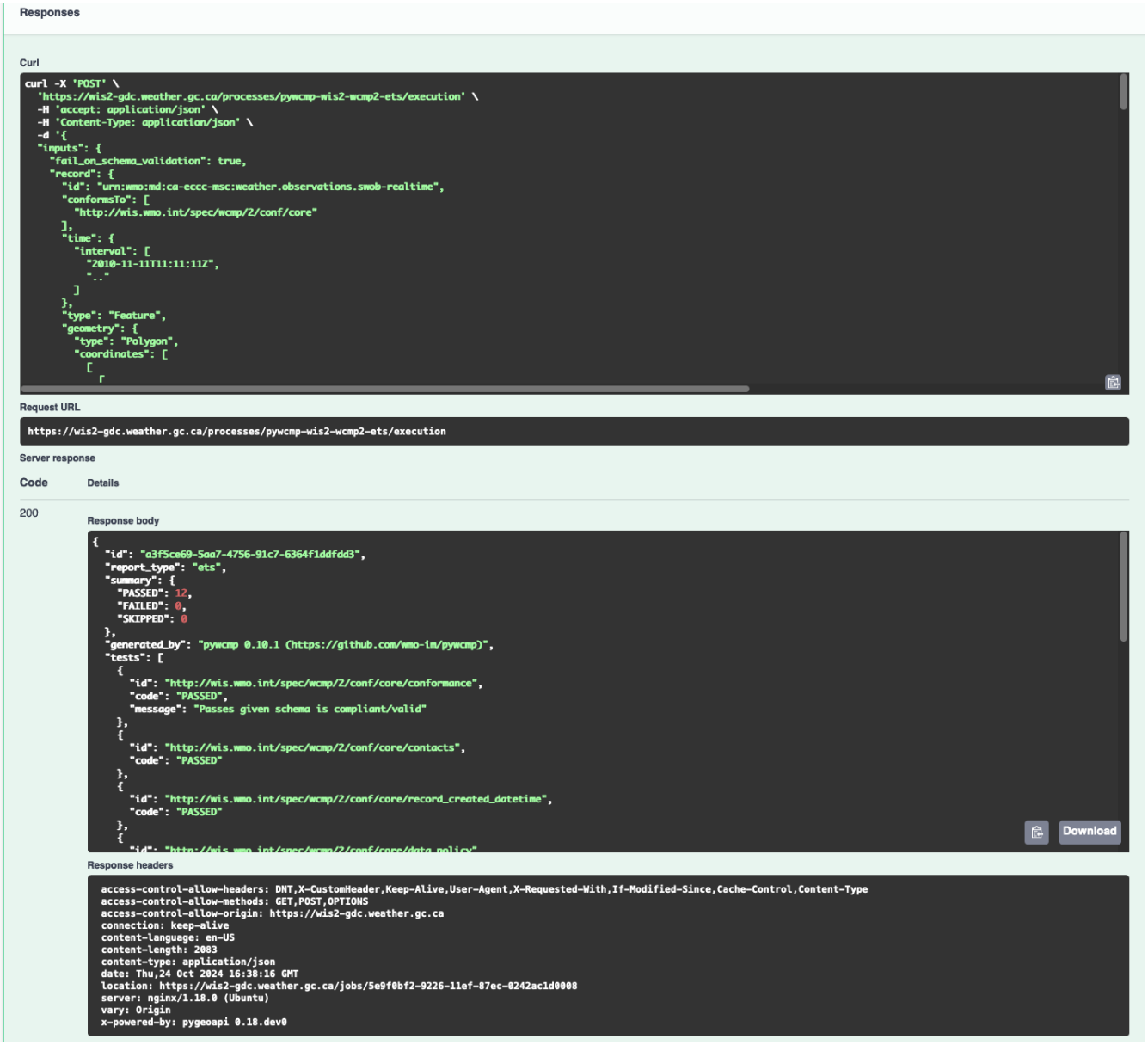

Figure 3. WIS2 GDC online validator, response

A response will be provided with validation results.

The WCMP2 standard also has (draft) Key Performance Indicators that provide quality assessment in support of continuous improvement of WCMP2 records. pywcmp additionally implements these KPIs using a similar workflow.

Using pywcmp for WCMP2 quality assessment

```
# validate WCMP2 against abstract test suite (file on disk)
pywcmp kpi validate /path/to/file.json

# validate WCMP2 against abstract test suite (URL)
pywcmp kpi validate https://example.org/path/to/file.json
```

The KPI check performs actions such as checking for healthy/working links and acronym checks, length of titles and more. Note that the KPI results do not affect WCMP2 compliance but are suggestions on how to improve your WCMP2 record for better discoverability and use.

The Canadian and German WIS2 Global Discovery Catalogues also provide this functionality online in a similar fashion as the validator.:

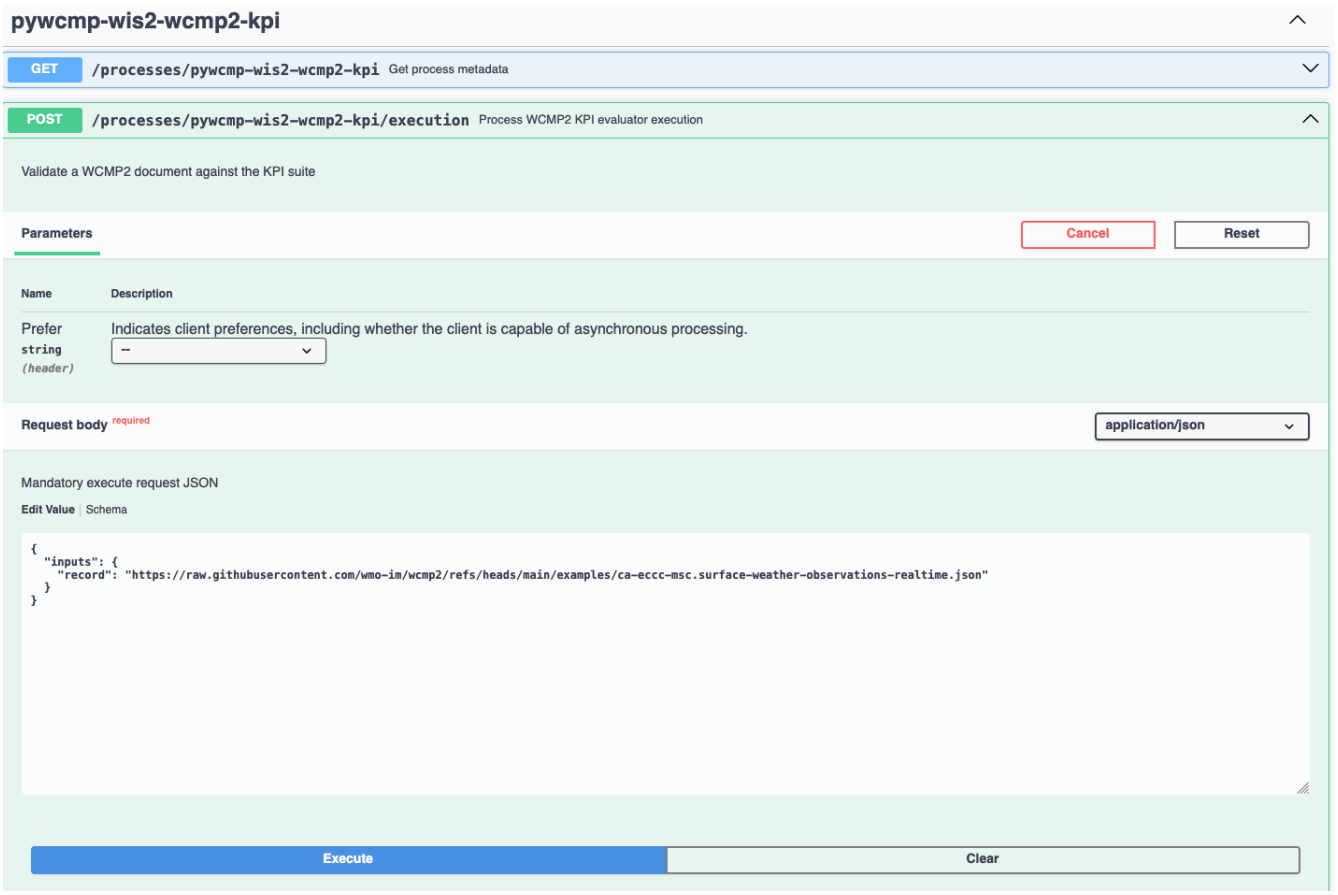

Figure 4. WIS2 GDC quality assessment checker, request via WCMP2 URL

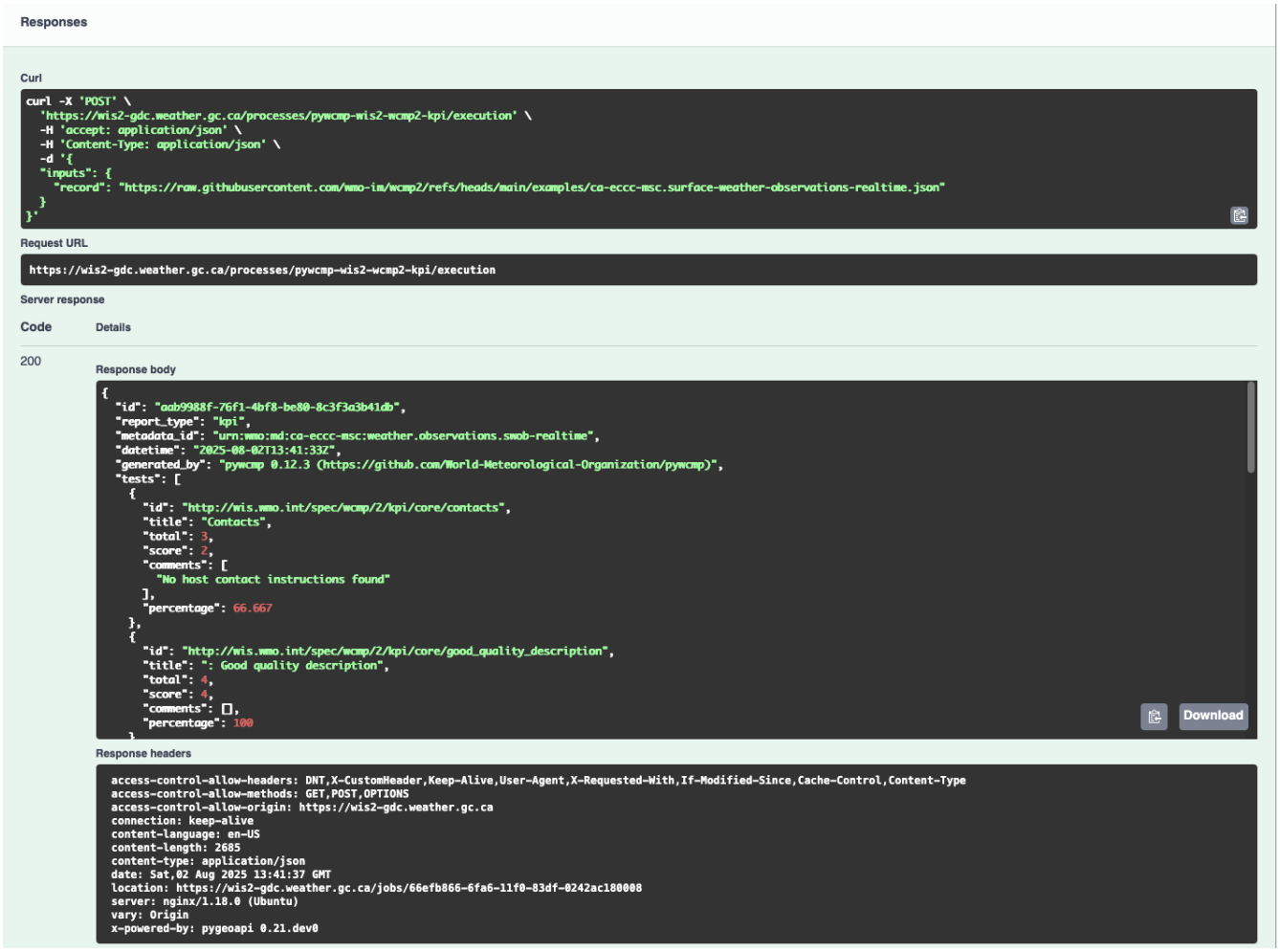

Figure 5. WIS2 GDC quality assessment checker, response

A response will be provided with quality assessment results (in a similar format to the validation report).

3.7. Subscribing to GDC reports of WCMP2 validation and quality assessment

When a WIS2 Node publishes a WCMP2 record, GDCs perform validation according to the WCMP2 Abstract Test Suite). If the record is compliant, a GDC will ingest and publish the record. In addition, a GDC may provide additional quality assessment testing based on the (draft) WCMP2 Key Performance Indicators (KPIs).

GDC reports are made available as part of the (draft) WIS2 Monitoring Events specification (WME) and can be found by subscribing to a Global Broker using the topic `monitor/a/wis2/CENTRE_ID`, where:

- `CENTRE_ID` is the centre identifier of the subject of the message

For example, to subscribe to GDC reports on WCMP2 records published by Japan Meteorological Agency (JMA), one would subscribe to the topic `monitor/a/wis2/jp-jma`.

Message payloads are based on WME Message Encoding (WMEM), with a `data` payload of the WCMP2 ETS/KPI reports.

3.8. Advertising client side filters for data subscriptions in WCMP2 and WNM

A key concept of a WCMP2 record is "actionable links"; this means being able to access a dataset or data granule without any further interactions. For real-time data, a WCMP2 record provides linkages to the WIS2 Global Broker via the MQTT protocol. At its core, MQTT has two key components:

- topic: the topic to subscribe to
- message payload: the message provided as part of a notification to a given topic

WIS2 defines the WIS2 Topic Hierarchy (WTH) and WIS2 Notification Message (WNM) standards which provide a standards-based GeoJSON payload/message.

A typical MQTT link in a WCMP2 document is defined as follows:

Typical WCMP2 MQTT link

```
{
  "rel"  : "items",
  "type" : "application/geo+json",
  "title": "WIS2 notification service",
  "href" : "mqtts://example.org",
  "channel": "cache/a/wis2/ca-eccc-msc/data/core/weather/surface-based-
observations/synop"
}
```

Given WCMP2, WTH and WNM, a user can subscribe to topics related to data of interest for download and access.

In some cases, a dataset may be organized in a manner which requires additional further "filtering" such that a data consumer is only interested in a certain subset of the data granules being advertised by a given WNM. Some examples include (but are not limited to), where a data consumer may be only be interested in:

- surface weather observations from a certain station, or
- numerical weather prediction forecast data for a certain timestep or weather parameter

To implement this behaviour, add additional properties to both WCMP2 and WNM as follows:

3.8.1. Example: Surface weather observations

Surface weather observations: WCMP2 MQTT link with additional properties

```
{
  "rel"  : "items",
  "type" : "application/geo+json",
  "title": "Real-time notifications",
```

```
  "href" : "mqtts://globalbroker.meteo.fr:8883",
  "channel": "cache/a/wis2/ca-eccc-msc/data/core/weather/surface-based-
observations/synop",
  "properties": {
    "wigos_station_identifier": {
       "type": "string",
       "title": "WIGOS station identifier"
    }
  }
}
```

Surface weather observations: WNM additional properties

```
{
  "properties": {
    "wigos_station_identifier": "0-20000-0-71628"
  }
}
```

When implemented by a data producer, a data consumer can:

- subscribe to real-time notifications to the given topic
- perform client side filtering by against all incoming WNMs with `properties.wigos_station_identifier = "0-20000-0-71628"`

3.8.2. Example: Numerical weather prediction based forecast

Numerical weather prediction: WCMP2 MQTT link with additional properties

```
{
  "rel"  : "items",
  "type" : "application/geo+json",
  "title": "Real-time notifications",
  "href" : "mqtts://globalbroker.meteo.fr:8883",
  "channel": "origin/a/wis2/ca-eccc-msc/data/core/weather/prediction/forecast/medium-
range/deterministic/global",
  "properties": {
    "model_run": {
       "type": "string",
       "title": "Model run",
       "enum": [
           "00",
           "12"
        ],
        "example": "00"
    },
    "forecast_hour": {
       "type": "string",
       "title": "Forecast hour",
```

```
        "example": "004"
    }
  }
}
```

Numerical weather prediction: WNM additional properties

```
{
  "properties": {
    "model_run": "00",
    "forecast_hour": "004"
  }
}
```

A data producer would extend WCMP2 and WNM as follows:

- WCMP2: add a link `properties` object for MQTT links, where each key of the link `properties` object is a JSON Schema property definition
- WNM: add additional properties (key: value pairs) in the `properties` object as desired

When implemented by a data producer, a data consumer can:

- subscribe to real-time notifications to the given topic
- perform client side filtering against all incoming WNMs with `properties.model_run = "00" and properties.forecast_hour = "004"`

A sample Python script can be found below. The script connects to the Météo-France Global Broker, subscribed to weather notifications from Environment and Climate Change Canada, Meteorological Service of Canada. The script then performs client side filtering by evaluating (for each WNM) the `properties.wigos_station_identifier` value to match a particular station (`0-20000-0-71628`).

Sample Python script to perform client side filtering

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

broker = 'globalbroker.meteo.fr'
port = 8883
username = 'everyone'
password = 'everyone'
topic = 'cache/a/wis2/ca-eccc-msc/data/core/weather/surface-based-observations/synop'

wsi_to_filter = '0-20000-0-71628'


def connect_mqtt() -> mqtt_client:
    def on_connect(client, userdata, flags, reason_code, properties):
        if reason_code == 0:
            print(f'Connected to {broker}')
```

```
        else:
            print(f'Failed to connect: {reason_code}')

    def on_log(client, userdata, level, message):
        print("LOG:", message)

    client = mqtt_client.Client(mqtt_client.CallbackAPIVersion.VERSION2,
                                client_id='s123')
    client.username_pw_set(username, password)
    client.on_connect = on_connect
    client.on_log = on_log
    client.tls_set(tls_version=2)
    client.connect(broker, port)

    return client


def subscribe(client: mqtt_client):
    def on_message(client, userdata, message):
        message_dict = json.loads(message.payload.decode())

        print('Performing client side filtering')
        wsi = message_dict['properties'].get('wigos_station_identifier')

        if wsi != wsi_to_filter:
            print(f'Topic: {message.topic}')
            print(f'Payload: {message.payload.decode()}')

    client.subscribe(topic)
    client.on_message = on_message


def run():
    client = connect_mqtt()
    subscribe(client)
    client.loop_forever()


if __name__ == '__main__':
    run()
```

3.9. Providing a requirements specification for a WIS 2.0 Node

The below recipe provides user requirements specifications for the implementation of a WIS 2.0 Node.

3.9.1. WIS 2.0 Node - User requirements specifications

3.9.1.1. Introduction

Applicable Documents

	Document Title	Reference
AD-0	Manual on the WMO Information System, Volume II - WMO Information System 2.0	https://library.wmo.int/idurl/4/68731

Reference Documents

	Document Title	Reference
RD-0	Guide to the WMO Information System Volume II - WMO Information System 2.0	https://library.wmo.int/idurl/4/69130
RD-1	Provisions for the Transition from the WMO Information System (WIS) 1.0 and Global Telecommunication System to WIS 2.0	https://library.wmo.int/idurl/4/69050
RD-2	WMO Information System 2.0 Strategy	https://library.wmo.int/doc_num.php?explnum_id=4620
RD-3	WIS 2.0: How to define successful transition?	https://wmo-teams.atlassian.net/wiki/spaces/WIS2/pages/301957121/WIS2.0+how+to+define+successful+Transition

3.9.1.2. System overview

System Context

The WIS 2.0 Node will be available to receive data from the upstream data production system, and to serve these data via the Internet to the WIS 2.0 environment.

The WIS 2.0 Node will be available to receive metadata and to serve these metadata via the Internet to the WIS 2.0 environment.

WIS 2.0 Global Brokers will subscribe to the WIS 2.0 Node, and will receive publication messages when data and metadata become available.

WIS 2.0 Node

1. Selected core and recommended data from upstream systems
2. Notification messages to WIS 2.0 Global Brokers
3. Selected core data to WIS 2.0 Global Caches
4. Subscription messages from WIS 2.0 Global Brokers
5. Remaining core data and recommended data to WIS 2.0 end users

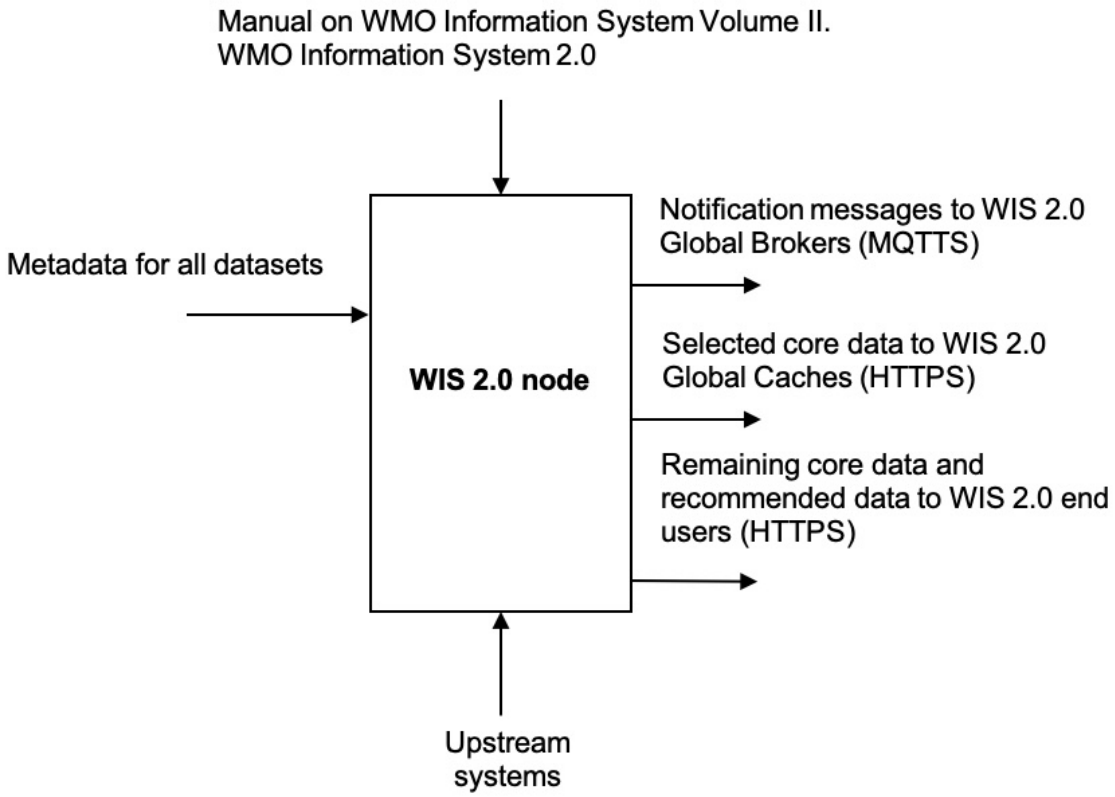

Figure 6. WIS 2.0 Node context diagram

External Interfaces

As identified in the context diagram above, the WIS 2.0 Node will have external interfaces with WIS 2.0 Global Brokers, and WIS 2.0 Global Caches. In each case, the communications will be via the Internet.

The Global Brokers will subscribe to the MQTT broker on the Node. This will be done using the standard MQTT secure port, 8883.

The Global Caches will retrieve core data from the Node by accessing the HTTP server on the Node.

End users will retrieve core data not retained by the Global Caches from the Node by accessing the HTTP server on the Node. End users will also retrieve recommended data from the Node by accessing the HTTP server on the Node, subject to appropriate access control.

The WIS 2.0 Node will be compliant with the Manual on WMO Information System Volume II. WMO Information System 2.0 [AD-0].

3.9.1.3. User requirements

The purpose of the WIS 2.0 Node (hereafter, referred to as the Node) is to be available to receive data from the production system, and to serve these data via the Internet to the WIS 2.0 environment.

In order to reach this target, the following high-level user requirements need to be fulfilled:

USR-0001

The Node shall comply with the specification given in the Manual on WMO Information System Volume II. WMO Information System 2.0, [AD-0].

USR-0101

The WIS 2.0 Node shall include an MQTT broker, using MQTT 5.0 (hereafter, referred to as the Broker).

USR-0150

The Broker included in the Node shall be accessible via MQTT protocol over the Internet.

USR-0201

The Node will allow subscriptions from the WIS 2.0 Global Brokers.

USR-0210

The Node will restrict subscriptions to the Broker by only WIS 2.0 Global Brokers by filtering their incoming IP addresses as made available by WMO Secretariat.

USR-0220

Access to the Broker shall be password controlled.

USR-230

The secure version of MQTT (MQTTS) shall be used. The use of SSL certificates to support this shall be maintained over the lifetime of the system.

USR-240

The Broker shall publish messages using MQTT's Quality of Service (QoS) level 1. This is defined as follows: "*The broker/client will deliver the message at least once, with confirmation required.*".

USR-0301

Upon the arrival of data for distribution via WIS 2.0, the Broker shall publish an MQTT message

announcing the availability of the data.

USR-0320

Notification messages published by the Node shall be formatted in geoJSON, in accordance with the Manual on WMO Information System Volume II. WMO Information System 2.0, [AD-0].

USR-0340

Notification messages published by the Node shall be published using an MQTT topic defined in accordance with the WIS 2.0 topic hierarchy.

USR-0360

The MQTT topic used in notification messages shall not be configured to retain messages.

USR-0401

The Node shall provide access to core data via HTTP over the Internet.

USR-0450

The Node shall provide access to recommended data via HTTP over the Internet, subject to appropriate access control.

USR-0500

The Node shall provide access to core data via HTTPS over the Internet.

USR-0550

The Node shall provide access to recommended data via HTTPS over the Internet, subject to appropriate access control.

USR-0601

The Node shall support the retrieval core data by the WIS 2.0 Global Caches.

Extract from the Manual on WMO Information System Volume II. WMO Information System 2.0 [AD-0]:

FUNCTIONAL REQUIREMENTS OF A WIS NODE

3.6.1 General

3.6.1.1 A WIS Node is the component that enables an NC or DCPC to publish their data and discovery metadata via WIS.

3.6.1.2 See also 3.3 (Functional requirements of an NC) and 3.4 (Functional requirements of a DCPC).

3.6.2 Provide access to data and discovery metadata

3.6.2.1 A WIS Node shall provide access to data in accordance with the WMO Unified Data Policy (Resolution 1 (Cg-Ext-2021)).

3.6.2.2 A WIS Node shall allow one or more Global Caches to access and download core data it publishes for real-time and near real-time exchange. Global Caches provide highly available access to copies of these resources.

3.6.2.3 A WIS Node may restrict access to its core data, relying on Global Caches providing access to data consumers.

3.6.2.4 A WIS Node may provide access to data using a Web-based Application Programming Interface (API).

3.6.2.5 A WIS Node shall provide access to discovery metadata describing the data it makes available and how that data can be accessed. Discovery metadata from a WIS Node is added to the Global Discovery Catalogue to create a consolidated view of data available from all WIS Nodes.

3.6.2.6 A WIS Node shall have the capability to publish notifications via a Message Broker.

3.6.2.7 A WIS Node shall publish notifications via its Message Broker about updates to the data and discovery metadata it provides – including the availability of new data, changes to discovery metadata, and removal of a data set from WIS.

3.6.2.8 A WIS Node shall use a standardized topic structure when publishing notifications. Note: More information on the standardized topic structure is provided in the Guidance on technical specifications of WIS 2.0.

3.6.2.9 A WIS Node shall allow one or more Global Brokers to subscribe to notifications published via its Message Broker. Global Brokers provide highly available distribution of notifications published by a WIS Node.

3.6.2.10 See also 4.3 (WIS-TechSpec-2: Publishing data and discovery metadata).

Note: More information on the function and implementation of a WIS Node is provided in the Guidance on technical specifications of WIS 2.0.

3.6.3 Monitor performance of a WIS Node

3.6.3.1 Each WIS Node shall contribute to monitoring the performance of WIS.

3.6.3.2 See also 4.7 (WIS-TechSpec-6: Managing operations of the WIS).

[1] https://www.go-fair.org/fair-principles/a2-metadata-accessible-even-data-no-longer-available/

[2] https://standards.iso.org/iso/19115/-3/mcc/1.0/codelists.html

Chapter 4. Recipes for Earth system discipline domain experts

4.1. Defining and proposing topic for WMO Earth system disciplines

The WIS2 Topic Hierarchy describes the topic structure and levels to be used when publishing WIS2 notification messages.

The purpose of this document is to provide guidelines to domain experts so that the Topic Hierarchy definition is consistent and useful to address the needs of WIS2 users.

4.1.1. Core levels

Core topics are defined in the first 7 levels and address all Earth system disciplines in a consistent manner.

Core topic examples:

- `cache/a/wis2/ma-marocmeteo/core/data/weather`
- `origin/a/wis2/de-dwd/core/data/ocean`
- `cache/a/wis2/jp-jma/core/data/weather`

The Earth system disciplines defined by WTH are as follows:

- `atmospheric-composition`
- `climate`
- `cryosphere`
- `hydrology`
- `ocean`
- `space-weather`
- `weather`

Topics within each Earth system discipline are then defined by domain experts, reviewed, approved, and published by WMO.

4.1.2. Additional Earth domain specific topics levels

4.1.2.1. What is the use of the Topic Hierarchy ?

The definition of the Topic Hierarchy is heavily linked to the use of Publish-Subscribe (Pub/Sub) protocols, here MQTT[S], in WIS2. Users, when subscribing to the Global Broker, can decide which notification messages they wish to receive.

4.1.2.2. MQTT wildcards

For example, connecting to a Global Broker and subscribing to:

`cache/a/wis2/+/core/data/weather/surface-weather-observation/synop`

users will receive a notification when a new synop is made available from **any** centre-id publishing message on WIS2. The `+` character is a single level wildcard for MQTT subscription.

A user can also choose to subscribe to:

`cache/a/wis2/de-dwd/core/data/weather/#`

In this case, users will receive notification messages from the `de-dwd` centre-id for **all** weather data. The `#` character is a multiple level wildcard for MQTT subscription, and can only be used at the end of a topic subscription.

By using `+` and `#` (the two defined wildcards in the MQTT[S] protocol), users can extend their subscription and receive all messages they need.

The purpose of additional topic hierarchy levels is to allow users to get the messages they need and to be more specific in their request.

4.1.3. Dos and don'ts when defining Earth system discipline topics

Define only a *small number* of additional levels

According the the MQTT[S] protocol specification, the accepted length of a given topic is more than 65kB. This means that the number of levels can be extremely large. However, as explained in https://www.emqx.com/en/blog/advanced-features-of-mqtt-topics:

Try not to use more topic levels "just because I can". For example, `my-home/room1/data` *is a better choice than* `my/home/room1/data`.

By default, some MQTT brokers are configured to accept a maximum of 10 levels. This can be changed in the configuration of the broker, however, this limit shows that a usable, practical topic structure should not be too deep (i.e. with a large number of levels).

For WIS2, and considering the various Earth system disciplines, a limit of **4** sublevels seems appropriate.

Do not use the topic as a metadata record

When defining the topic, experts must focus on the needs of users. The purpose of the WIS2 Topic Hierarchy is **not** to describe as precisely as possible what data users will obtain if they decide to subscribe and then download. Rather, it is *only* to provide a filtering mechanism so that users will not be flooded by WIS2 Notification Messages that may not be useful for them.

In WIS2, all datasets **must** be described using the WMO Core Metadata Profile version 2(WCMP2). Users will be able to discover the data they need by searching the WIS2 catalogue using (via search engines, directly, or from portals and applications). The topic hierarchy information will be part of the metadata record for data which provides real-time notifications of publication.

*Do not allow locally defined sublevels outside the `experimental` topic

Each Earth system discipline provides an `experimental` topic. For example, for `weather`, the first additional levels in the domain topic hierarchy are:

Name	Description
`advisories-warnings`	Advisories and warnings
`aviation`	Aviation
`prediction`	Data sets produced by quantitative algorithms, such as numerical or statistical prediction models, describing the past, present and future meteorological states
`space-based-observations`	Space based observations
`surface-based-observations`	Surface based observations
`experimental`	Experimental topics

As the name suggests, `experimental` allows for defining additional topics for tests and experiments. This is not meant to be used for operational data exchange. It should only be used for testing purposes.

With the exclusion of the `experimental` topic level, the topic Hierarchy must be *fully* defined for each Earth system discipline.

In some situations, it might be tempting for a data producer to use additional topic levels to restrict even more the number of messages received by the users.

This is **forbidden**.

In each Earth system discipline community, all WIS Centres will be able to use the entire topic hierarchy of the domain if they provide data corresponding to each topic. A WIS Centre will not be allowed to add additional sublevels or undefined level within the Topic Hierarchy for its own needs.

Modification of the Topic Hierarchy will be possible by using the WMO fast-track approval process.

Consider users needs and prevent complex wildcard subscriptions

The purpose of the WIS2 Topic Hierarchy is to inform users about the availability of new data. In WIS2, obtaining data will start, in most cases, by configuring one or more subscription to topics, as defined in the associated WCMP2 discovery metadata records, so that users will receive notifications when new data is available.

The Topic Hierarchy should be defined so that users will not need to configure a very large number of different subscriptions to get the data they are interested in.

Each level in the WIS2 Topic Hierarchy should be seen as a "logical" group (as the Earth system disciplines `weather`, `ocean`... or like `synop` for `surface-based-observations`).

Then, and considering that wildcard subscription (using `+` and `#` as described above), are "expensive" to manage for the brokers.

For example, a topic hierarchy resulting in users subscribing to:

```
cache/a/wis2/+/core/data/ocean/+/some/+/thing/+/else/#
```

should be avoided. A subscription to the following topic:

```
cache/a/wis2/+/core/data/ocean/some/thing/else/#
```

is much more effective for both the client and the broker side.

If it is likely that most users will use wildcards for particular topic levels, then, either removing that level altogether, of moving that level a the end of the topic hierarchy is also more efficient for clients and producers.

If most users end up subscribing to:

```
cache/a/wis2/+/core/data/ocean/+/thing/+/#
```

then, the Topic Hierarchy could be reconsidered, so that the above subscription can be replaced by:

```
cache/a/wis2/+/core/data/ocean/thing/#
```

Reordering the levels of topics and potentially reducing the number of sublevels makes the topic hierarchy simpler and more efficient.

Facilitate client side filtering

Notification messages are small pieces of information. MQTT[S] broker and clients are able to handle a very large number of messages. In that sense, receiving, potentially, too many messages is not a problem. However, downloading data, depending on the size of the data might be slower and less efficient. If, for a particular dataset, the geometry information available in the notification message is not sufficient to allow client-side filtering before download, it is suggested to provide additional information in the `properties` object of the notification message so that users can decide *before* downloading if the data in this particular messsage is useful for them.

See [_advertise_client_side_filters_for_data_subscriptions_in_wcmp2_and_wnm] for more information on client side filtering

4.2. WCMP2 Extensions: creating Earth system discipline domain specific profiles of WCMP2

OGC API - Records - Part 1: Core: Requirements Class: Record Core allows for extension and profiling of the metadata content model. In this context, WCMP2 itself is a domain profile, as WMO's discovery metadata standard for all Earth system disciplines.

As part of WIS2, Earth system discipline domain experts may choose to extend and profile WCMP2 to meet their specific needs. Typical examples of where extensions can be useful include, but are not limited to:

- further constraining specific WCMP2 properties (e.g. based on a controlled vocabulary)
- enforcing optional WCMP2 properties to be required
- adding new or additional properties to a WCMP2 record

Profiling WCMP2 can enable domain communities achieve deeper interoperability (or tighter coupling) to meet their needs. For example, an NWP profile of WCMP2 may drive a user-focused NWP portal which leverages domain specific properties of a WCMP2 record for more meaningful search results and assessment for use.

4.2.1. Considerations for creating a WCMP2 Extension

Before creating an extension, domain experts need to give thought on the following aspects:

- gap analysis in WCMP2: can existing WCMP2 properties be used?
- use cases: consider how the additional properties (or constraints on existing) would be used (as a queryable, returnable, etc.)
- sustainability: while developing a WCMP2 extension can be worthwhile metadata modelling exercise, ensure that there is a team in place to help maintain the extension over time. This means having clear ownership of a WCMP2 extension that is subject to ongoing review, and supporting user questions, issues, enhancements and bug fixes

4.2.2. How to create a WCMP2 Extension

WCMP2 extensions can be defined on WMO's GitHub repository for WCMP2 Extensions. Domains can identify and propose extensions as GitHub issues and Pull Requests, which are reviewed by WMO TT-WISMD:

The following elements are required for any WCMP2 extension:

- directory on GitHub repository, containing:
  - examples (directory `examples/`): a directory of example WCMP2 records exemplifying the extension
  - JSON Schema (directory `schema/`): a directory of the JSON Schema definition for the extension, encoded as YAML. The JSON Schema definition should either refer to controlled vocabularies from existing online vocabularies or define inline as `enum` lists
  - README (file `README.md`): the WCMP2 Extension Specification, which includes:
    - owner: clear, unambiguous ownership identification of the Extension
    - prefix: prefix for all defined elements (i.e. `hydro:`, `nwp:`, etc.)
    - conformance: URI for conformance identification and usage in WCMP2 documents (`conformsTo` member)
    - maturity: level of maturity guideline for usage
    - definitions: specification of additional elements and their requirements (required/optional)

Example directory structure

```
|__hydro
|____README.md
|____schema
| |____hydro-schema.yaml
|____examples
| |____example1.json
| |____example2.json
```

An example extension can be found at https://github.com/wmo-im/wcmp2-extensions/tree/main/example.

4.3. Differences in WCMP2 records with Extensions

A WCMP2 record with a defined extension per above would provide the following in `conformsTo`:

Example of identifying additional conformance for a WCMP2 Extension

```
"conformsTo": [
    "http://wis.wmo.int/spec/wcmp/2/conf/core",
    "https://wmo-im.github.io/wcmp2-extensions/hydro"  # TODO: clarify how to best
identify
]
```

This will allow a WCMP2 parser to detect additional conformance to a given Extension and validate accordingly (in addition to validating WCMP2 Core).

Chapter 5. Recipes for Global Service operators