ES-DOC - ESGF Publisher Integration

Overview **2**

Implementation **2**

Software Components 2

The ESG Publisher 2

cdf2cim 2

Installing cdf2cim 3

Integration between the ESG Publisher and cdf2cim 3

Calling the ESG Publisher with cdf2cim enabled 3

Where does cdf2cim do with the data it finds? 4

Instructions to Data Node Managers 4

Setting up authorisation access to the ES-DOC web service 4

Request access to the “ES-DOC-INSTITUTIONAL” GitHub team 4

Create an access token from GitHub 5

Set your web service environment variables 5

Test that the credentials are valid 6

The recommended ESG Publisher workflow in relation to ES-DOC 6

Potential housekeeping issues 7

Issues and edge-cases **7**

Notes from discussions with ESGF colleagues **7**

Appendices **8**

Appendix 1: Example JSON content for a CMIP5 record 8

Appendix 2: Example JSON content for a test CMIP6 record 8

Overview

This document provides information about the integration between ES-DOC and ESG Publisher. The particular integration is between the **`cdf2cim`** tool, that scans the content of NetCDF files, and the command line **`esg-publisher`**. The interactions are explained and then details are provided of how Data Node Managers need to set things up in order for the integration to work. The `cdf2cim` tool can be accessed as a configurable component of the `esgpublish` command and is included as a default action when publishing CMIP6 data sets. Once the authentication components have been configured then the act of publication should automatically call `cdf2cim` and push relevant metadata to the ES-DOC web service.

Implementation

Software Components

The ESG Publisher

The ESG Publisher, part of ESGF, scans netCDF metadata to enable data discovery, metadata interrogation, data download and sub-setting at both the file and dataset level. It captures information from the netCDF data files to generate aggregations and metadata summaries suitable for publishing to various sources, including the THREDDS data server and the ESGF Search system. The metadata extracted during this process is used to underpin the user interfaces that allow interrogation and access at both the file and dataset levels.

The ESG Publisher is available at:

https://github.com/ESGF/esg-publisher

cdf2cim

The Earth System Documentation (ES-DOC) ecosystem is able to capture, describe and disseminate essential information about climate modelling activities. Within CMIP6, scientists are describing their models and experiments in detail using a rich semantic model known as CIM2 (Common Information Model 2). ES-DOC also requires information about ensemble runs and each individual simulation. A *simulation* document is an independent entity, but all *simulations* are considered to be part of an *ensemble*. The ES-DOC tools (https://es-doc.org) allow users to select and compare simulations. A CIM2 simulation document describes a single integration by a particular model and why the integration was carried out. Time span, ensemble and parent simulation properties are also described.

For CMIP5, simulations had to be documented manually. As a result, many were undocumented and there were often errors in records that were produced. The extensive global metadata in CMIP6 netCDF data files provides sufficient information to allow the simulation content to be extracted by scanning the files directly. Automation has the potential to ensure that everything is completely documented and based on the actual archived data files.

A command-line tool and Python library, **cdf2cim**, has been developed to manage the file scanning, serialisation to JSON, and upload to the ES-DOC server. cdf2cim is packaged so that it can be imported by the ESG Publisher. The cdf2cim code is available from:

https://github.com/ES-DOC/esdoc-cdf2cim

Installing cdf2cim

The standard installation of the ESGF Data Node includes installation of both the esg-publisher and the cdf2cim library. If you are installing the esg-publisher in stand-alone mode then the cdf2cim library will be installed when you resolve the dependencies using an appropriate package manager (such as pip). The cdf2cim dependencies are described in the requirements file at:

https://github.com/ES-DOC/esdoc-cdf2cim/blob/master/requirements.txt

Typically cdf2cim can be “pip” installed:

```
$ pip install cdf2cim
```

Integration between the ESG Publisher and cdf2cim

The following location in the esg-publisher code provides the interface with the cdf2cim library:

https://github.com/ESGF/esg-publisher/blob/master/src/python/esgcet/esgcet/publish/cim.py

The actual call out to cdf2cim is here:

https://github.com/ESGF/esg-publisher/blob/master/src/python/esgcet/esgcet/publish/cim.py#L92

The input to cdf2cim is a single directory. It finds all netCDF inside the directory (and any sub-directories) and scans them for CIM2 *simulation* content. It reduces the content into a single JSON record.

Calling the ESG Publisher with cdf2cim enabled

The cdf2cim library is enabled using ”`create_cim=True`” setting in the esg.<project>.ini file. This is set by default for CMIP6.

If you wish to include the call to cdf2cim when running for other projects, then you will need to add the line above, or you can explicitly call it with one of the following arguments to the `esgpublish` command:

```
--create-cim
```

will run cdf2cim as the first step of a normal publication. If it fails it will issue a warning and carry on. This flag **must** be used in combination with `--map` (which will get the publisher to scan a mapfile to find out which datasets/files should be published). We can assume that `--map` will be run. If `--map` is not used when `--create-cim` is used the publisher will raise an exception.

```
--create-cim-only
```

will run cdf2cim and nothing else (i.e. nothing else in the ESG publication process). If it fails it will issue a warning and carry on. This flag **must** be used in combination with `--map` (which will get the publisher to scan a mapfile to find out which datasets/files should be published). We can assume that `--map` will be run. **This option is used if you want to run separately from the rest of the publisher actions - it will be appropriate for separating out the workflow if/when necessary (e.g. if you missed a set of data sets from cim-generation and wanted to go back and run them).**

Additional command-line arguments that may be useful are: “`--no-create-cim`” (do not call cd2fcim) and “`--verbose-cim-errors`” (in the event of failure to publish CIM documents, display server errors).

Where does cdf2cim do with the data it finds?

Each *simulation* record generated from a collection of netCDF files is stored in a local JSON file in the directory:

```
$HOME/.esdoc/cdf2cim/scanned/
```

When called by the ESG Publisher, cdf2cim will attempt to upload the contents of any JSON files found in that directory to the ES-DOC server. After successful upload the JSON files will be moved to:

```
$HOME/.esdoc/cdf2cim/published/
```

The size of each simulation record is very small (O(1kB)) so this should not impact on the Data Node or Publisher installation/service.

The default behaviour when used with the ESG Publisher is:

1. The Publisher makes a single call to `cdf2cim` per ESGF Dataset.
2. A JSON “blob” is created in memory.
3. Log input directory and hash-based blob file name (in cdf2cim log).
4. If found in “published/” directory:
    a. Take no action.
    b. Log: no action taken.
5. Else:
    a. If blob not found in “scanned/” directory:
        i. Blob written to “scanned/” directory
    b. Default behaviour is that the publish action will push ALL blobs in “scanned/” directory to server.
    c. If success:
        i. Move blobs to “published” directory
        ii. Log: success
    d. If failed:
        i. Leave in “scanned” directory
        ii. Log: error

Instructions to Data Node Managers

Setting up authorisation access to the ES-DOC web service

The ES-DOC server restricts access to users who are part of the `cdf2cim-publication` GitHub team. This is simply a list of GitHub accounts grouped for a common purpose (i.e. ES-DOC publishing). **Data Node Managers will need to request access to the GitHub team in order to be able to send content to the ES-DOC web service.**

Request access to the “ES-DOC-INSTITUTIONAL” GitHub team

In order to register for this team please e-mail cdf2cim@es-doc.org and with the following request:

Subject: *Request for access to ES-DOC-INSTITUTIONAL GitHub team*

Message:

Dear CDF2CIM Admin,

Please can you add me to the ES-DOC-INSTITUTIONAL GitHub team. I am working on the <INSERT YOUR ORGANISATION> ESGF Data Node and I am responsible for data publishing to CMIP6.

My GitHub ID is: <INSERT YOUR GITHUB ID>

Kind regards,

<YOUR NAME>

You should receive an automated response that looks like:

Dear CDF2CIM user,

Thank you for your ES-DOC-INSTITUTIONAL publication help request. It will be processed by a member of the ES-DOC team as soon as possible.

Best Regards,

The ES-DOC team

Once you have been informed that you are a member of the `ES-DOC-INSTITUTIONAL` GitHub team then you will be able to view the team membership here:

https://github.com/orgs/ES-DOC-INSTITUTIONAL/teams/cdf2cim-publication

Create an access token from GitHub

In order to publish content to the ES-DOC web service you need to provide the following details:

- Web service end-point (URL)
- GitHub account name
- GitHub access token

Please follow the instructions here to set up your access token. (More detailed information can be found here: https://help.github.com/articles/creating-a-personal-access-token-for-the-command-line/).

1. Login to the GitHub website using your account:
   https://github.com/login
2. In the upper-right corner of any page, click your profile photo, then click **Settings**.
3. Click on **Developer Settings** in the left-hand menu.
4. In the left-hand menu, click **Personal access tokens**.
5. Click **Generate new token**.
6. Give your token the name “cdf2cim” (in **Token description**).
7. Select the **read:org** scope for your token.
8. Click **Generate token**.
9. **Copy the token** to your clipboard. For security reasons, after you navigate off the page, you will not be able to see the token again.
10. Keep your token somewhere safe. Treat it like a password.

Set your web service environment variables

The following environment variables need to be set in your session before you attempt to run cdf2cim. You could set them all in a configuration file or setup file. E.g.:

```
# Environment variable: web-service host (optional)
export CDF2CIM_CLIENT_WS_HOST=https://test-cdf2cim.es-doc.org

# NOTE: For production system use: https://cdf2cim.es-doc.org

# Environment variable: GitHub user.
export CDF2CIM_CLIENT_GITHUB_USER=__INSERT_YOUR_GITHUB_USERID_HERE__

# Environment variable: GitHub access token.
export CDF2CIM_CLIENT_GITHUB_ACCESS_TOKEN=__INSERT_YOUR_GITHUB_ACCESS_TOKEN_HERE__
```

Test that the credentials are valid

It is useful to test that your access credentials are all working. The following web-service call to test that your credentials are working. Insert your GitHub user ID and access token into this URL and paste it into your browser:

https://test-cdf2cim-api.es-doc.org/verify-authorization?login=<GITHUB_USER_ID>&token=<TOKEN>

If your credentials are accepted you should receive a response that looks something like:

```
{"message": "ES-DOC CDF2CIM publication membership is active", "version": "0.2.0.0"}
```

The recommended ESG Publisher workflow in relation to ES-DOC

We recognise that within CMIP6 it is very important that information about simulations and ensembles is captured in the ES-DOC (Earth System Documentation: https://es-doc.org) ecosystem. In order to do this we need the esg-publisher to scan all netCDF files for core metadata (as defined in this schema:

https://github.com/ES-DOC/esdoc-cdf2cim-ws/blob/master/cdf2cim_ws/schemas/body/1.cmip6.json).

The package that publishes this information to ES-DOC is known as **cdf2cim** (https://github.com/ES-DOC/esdoc-cdf2cim). The cdf2cim python package is installed as part of the standard esg-publisher installation. Whenever you run the `esgpublish` command for **CMIP6** the call to cd2cim (unless you set the command-line flag `--no-create-cim`) will automatically scan the netCDF files and publish the ES-DOC content. (NOTE: for other (non-CMIP6) projects there is no automatic generation of ES-DOC content. This work happens inside the CMIP6 Handler).

The cdf2cim workflow is as follows:

1. Before the normal publisher actions are performed the `cdf2cim` library is called for each dataset.
2. One or more netCDF files are scanned and the common information found within the dataset is stored in a JSON file saved in: `$HOME/.esdoc/cdf2cim/scanned/`
3. The `cdf2cim` library will then make a request to the cdf2cim web service to push the contents of all JSON files (in: `$HOME/.esdoc/cdf2cim/scanned/`) up to the server.
4. If the JSON content is successfully added to the server:
    a. The JSON file(s) will be moved to: `$HOME/.esdoc/cdf2cim/published/`
5. If the JSON content FAILS to be added to the server:
    a. The JSON file(s) will remain in: `$HOME/.esdoc/cdf2cim/scanned/`
    b. Note that `esgpublish` will not fail. **The rest of the ESGF publication process will run regardless of the result of the ES-DOC interactions.**
    c. The JSON content might not be uploaded for a number of reasons:
        i. Your credentials are incorrectly defined.
        ii. You do not have the correct credentials.

   iii. The web service is unavailable.
   iv. The JSON document failed validation against the schema.

Once JSON documents have been pushed to the ES-DOC web service they will initially be cached on the file system of the server. A separate process will read the JSON documents on the ES-DOC server and add the content into the official CMIP6 document repository. At this stage the *simulation* and *ensemble* documents will be available via the ES-DOC search/web interface.

Potential housekeeping issues

ESGF Node Managers should be aware of the following situations that might require investigation:

1. If there are problems with the metadata in the netCDF files (i.e. they do not follow CMOR3/CMIP6 rules) then the ES-DOC server might reject the JSON content. Over time, this **could lead to a build up of (small) JSON files in the directory:** `$HOME/.esdoc/cdf2cim/scanned/`
   a. Whilst it is very unlikely that this would cause any issues (since the files are small) it would mean that the affected datasets have not been published to ES-DOC so the relevant simulation/ensemble records will not be available.
   b. If you identify this issue then please **contact the ES-DOC team** (cdf2cim@es-doc.org) to help diagnose the problem.

NOTE: we recommend that you leave the contents of the `$HOME/.esdoc/cdf2cim/published/` directory.

Issues and edge-cases

1. How to deal with retractions that potentially REMOVE content from existing CIM docs?
2. Raise issue with ESGF-PWT about $HOME/.esdoc/ directory hard-coded as place for Errata and CIM info to be cached. Would this cause a problem for anyone?

Notes from discussions with ESGF colleagues

1. We agreed that the call to cdf2cim automatically handled inside the CMIP6 Handler in the publisher. This means that everyone has to run it and it will not get missed.
   a. ESGF-PWT agree with this.
   b. We will also allow a `--no-create-cim` option to override the behaviour of the Handler.
   c. If create-cim FAILS (e.g. if Node Manager hasn't set up environment variables or joined team):
      i. Show them a warning and a pointer to the documentation.
   d. Using `--set-replica` would turn this default behaviour off (equivalent to setting `--no-create-cim`) - because replicas will already have been documented in ES-DOC.
   e. THIS IS IMPLEMENTED IN THE CMIP6 Handler:
      i. create_cim = True
2. Normal use case would be to put **environment variables** into the `esg.ini` file.
3. Where should the environment variables be set:
   a. In the publisher, we could say:
      i. If env variables are not already set:
         1. Then get from `esg.ini` config.
4. Do GitHub access tokens expire?
   a. No: this is under your control when you create the token. The default is no expiry.
   b. ...so one token could be used for the whole of CMIP6.
5. Can you change the directory in which the JSON files will be stored?

a. Yes, you can modify the directory by setting the environment variable: **CDF2CIM_CLIENT_IO_DIR**
b. This variable defaults to: `$HOME/.esdoc`

Appendices

Appendix 1: Example JSON content for a CMIP5 record

The following is a JSON record generated by scanning actual CMIP5 NetCDF files.

```
{
        "branch_time_in_parent": 0.0,
        "calendar": "noleap",
        "contact": "ipsl-cmip5 _at_ ipsl.jussieu.fr Data manager : Sebastien Denvil",
        "end_time": "2009-01-01 00:00:00",
        "experiment_id": "amip",
        "forcing": "Nat,Ant,GHG,SA,Oz,LU,SS,Ds,BC,MD,OC,AA",
        "initialization_index": 1,
        "institution_id": "IPSL",
        "mip_era": "CMIP5",
        "parent_experiment_id": "N/A",
        "physics_index": 1,
        "realization_index": 1,
        "references": "Model documentation and further reference available :http://icmc.ipsl.fr",
        "source": null,
        "source_id": "IPSL-CM5B-LR",
        "start_time": "1979-01-01 00:00:00"
}
```

Appendix 2: Example JSON content for a test CMIP6 record

The following is a prototype JSON record generated by scanning example CMIP6 NetCDF files.

```
{
    "activity_id": "CMIP",
    "branch_time_in_child": "1860-01-01T00:00:00Z",
    "branch_time_in_parent": "1809-11-12T00:00:00Z",
    "calendar": "360_day",
    "contact": "Python Coder (python@a.b.com) ",
    "end_time": "2006-06-01T00:00:00Z",
    "experiment_id": "piControl",
    "forcing_index": 1,
    "further_info_url": "something_or_other",
    "initialization_index": 1,
    "institution_id": "CSIRO-BOM",
    "mip_era": "CMIP6",
    "parent_forcing_index": 3,
    "parent_initialization_index": 1,
    "parent_physics_index": 1,
    "parent_realization_index": 1,
    "physics_index": 1,
    "realization_index": 1,
    "references": "Model described by Koder and Tolkien (J. Geophys. Res.,2001, 576 - 591).”,
    "source_id": "NICAM:",
    "start_time": "2005-12-01T00:00:00Z",
    "sub_experiment_id": "none",
    "variant_info": "forcing: black carbon aerosol only"
}
```