# EUMETSAT WP FCIDECOMP - Solution design

EUMETSAT

Nov 10, 2021

# LIST OF TABLES

# CONTENTS:

CHAPTER

ONE

# DOCUMENT INFORMATION

| | |
|---|---|
| Doc. id: | |
| External version: | |
| Author: | M. Cucchi (B-Open Solutions) M. Bottaccio (B-Open Solutions) |

# CHAPTER TWO

# INTRODUCTION

## 2.1 Purpose

The document describes a design proposal for a maintainable solution allowing users to reliably decode FCI L1c products compressed with CharLS.

## 2.2 Reference Documents

Table 2.1: Reference documents

| # | Title | Reference |
|---|---|---|
| [CONDA_VARIANTS] | conda-build – Build variants | https://docs.conda.io/projects/conda-build/en/latest/resources/variants.html |
| [FCIDE-COMP_CONDA] | FCIDECOMP Conda recipe developed by Martin Raspaud (SMHI) | https://github.com/mraspaud/fcidecomp-conda-recipe/ |
| [FCIDE-COMP_LATEST] | FCIDECOMP v1.0.2 repository | https://sftp.eumetsat.int/public/folder/UsCVknVOOkSyCdgpMimJNQ/User-Materials/Test-Data/MTG/MTG_FCI_L1C_Enhanced-NonN_TD-272_May2020/FCI_Decompression_Software_V1.0.2/EUMETSAT-FCIDECOMP_V1.0.2.tar.gz |
| [FCIDE-COMP_WPD] | Work Package Description | EUM/SEP/WPD/21/1244304 |
| [HDF5PLUGIN] | `hdf5plugin` python package | https://github.com/silx-kit/hdf5plugin |
| [HDFVIEW] | HDFView Software | https://www.hdfgroup.org/downloads/hdfview/ |
| [NETCDF_JAVA] | Unidata - NetCDF-Java | https://www.unidata.ucar.edu/software/netcdf-java/ |
| [NETCDF_JAVA_GITHUB] | NetCDF-C for reading (nj22Config.xml) in non-Unidata netCDF-Java based tools | https://github.com/Unidata/thredds/issues/1063 |
| [PANOPLY] | Panoply netCDF, HDF and GRIB Data Viewer | https://www.giss.nasa.gov/tools/panoply/ |

CHAPTER

# THREE

# CREATION OF CANONICAL REPOSITORY

The first step of the solution design consists in the creation of a single canonical repository, at EUMETSAT GitLab, hosting the official version of the FCIDECOMP software. This repository will contain all the code necessary to build the software and make it available as a HDF dynamically loaded filter.

## 3.1 Repository initialization

Initially, FCIDECOMP v1.0.2 (taken from *EUMETSAT sFTP repository*) will be taken as blueprint for the development of the solution codebase.

The repository will be put under configuration control. A new minor release adding README, BUILD, INSTALL, and LICENCE files, starting the Changelog, codifying the use of semantic versioning for future versions and adding a standardised build system will then be published.

## 3.2 Test suite

An initial test suite (at least against nominal conditions) will be implemented following the V&V strategy defined in [add reference]. As an initial proposal, we would implement most or all the tests as automated tests against the Python interface.

## 3.3 Test data

A preliminary set of test data taken from the MTG FCI L1C test data will be added to ensure a consistent and permanent dataset to execute tests.

# CHAPTER FOUR

# SUPPORT AND INTEGRATION WITH EXTERNAL TOOLS

In the following paragraphs outlines of the strategy to ensure that the FCIDECOMP software support all the required systems and usage patterns are reported.

## 4.1 Integration with HDF tools based on netCDF-C

The current implementation of the FCIDECOMP software (*v1.0.2*) already satisfies the HDF5 filters interface. So, once installed, it can easily be invoked by utilities relying on the `netcdf-c` library (such as `nccopy`), provided that:

- the location of the filter is specified in a specific environment variable, `HDF5_PLUGIN_PATH`;
- the correct filter id, if required by the utility, is specified (32018 for FCIDECOMP);
- any other parameter requested by the utility is also specified.

In order to provide a baseline support for CLI usage of the FCIDECOMP software, we will focus on its integration with `nccopy`. To foster such integration, we will:

- have the FCIDECOMP software package install the filter's library to a specific path at installation
- have the `HDF5_PLUGIN_PATH` environment variable automatically set each time a conda environment where FCIDECOMP is installed get activated
- document how to call `nccopy` to decompress files using the FCIDECOMP filter

## 4.2 Integration with Python

HDF5 filters are typically used in Python programs by invoking the `h5py` library, which may in turn require additional packages to provide specific filters. An example of such additional packages is the `hdf5plugin` library, which makes the FCIDECOMP filter, among others, available to `h5py`.

To grant integration with Python we will develop a package specific for the FCIDECOMP filter, initially as a stripped-down version of *hdf5plugin*. We will also try to contact maintainers of the `hdf5plugin` package to evaluate synergies and reduce conflicts. In this direction, Initial proposal is to have a small package which only includes the FCIDECOMP plugin support from `hdf5plugin`, and to propose the `hdf5plugin` maintainers to use it as a sub-module dependency in `hdf5plugin` in order not to break their interfaces and not to duplicate effort; in this case, this small python package would be maintained by B-Open (for the duration of the contract) on behalf of EUMETSAT.

## 4.3 Integration with EUMETSAT Data-Tailor

At the moment, the Data Tailor supports reading compressed FCI L1C products through the optional `epct_plugin_mtg4africa` customisation plugin, which in turns install FCIDECOMP by installing with `pip` the `hdf5plugin` package. Note that the same Data Tailor plugin also uses the FCICOMP software to compress output data, and that such compressor is built when building the Data Tailor plugin package. This has potential fo dependency conflicts, as the compressor and te decompressor rely in part on the same dependencies.

The approach to integrate the described solution with the Data Tailor will therefore include a revision of the current build and installation approach for the `epct_plugin_mtg4africa` customisation plugin. The most promising direction is having the plugin install FCIDECOMP from the conda package, so that dependency conflicts are to some extent managed by conda.

## 4.4 Integration with tools based on netCDF-Java

Use of the FCIDECOMP decompressor in Java may be complex to address. There is evidence that Unidata netCDF-Java based tools may work with HDF5 dynamic filters, but it is less clear whether they can still be used with non-Unidata applications based on NetCDF-Java (e.g. panoply). The main problem is related to the fact NetCDF-Java is essentially independent from HDF-Java (while NetCDF-C is based on HDF-C), and while HDF-Java can use HDF dynamically loaded filters as NetCDF-C based applications do, NetCDF-Java cannot.

A promising solution (see this *github issue*) consists in instructing NetCDF-Java based applications to read files using the NetCDF-C library, via a simple tweak to a configuration file. A preliminary test showed that, in the considered environment, the described solution is able to let *Panoply* (one of the most used NetCDF-Java based applications) decompress NetCDFs using the FCIDECOMP plugin. We will explore how the solution behaves with other notable NetCDF-Java based applications, such as *HDFView* and *ToolsUI*.

In the case this solution is adopted, it will only require to be documented as an effective work-around to use the FCIDECOMP plugin with NetCDF-Java based applications.

# CHAPTER FIVE

# PACKAGING STRATEGY

In the following paragraphs outlines of the strategy to build and package the FCIDECOMP software in order to ensure support for and integration with all the required systems are reported.

## 5.1 Supported platforms

Based on the *Work Package Description* and not considering OS that already have reached their End of Life, the FCIDECOMP software will support the following platforms:

- Windows 10, 32 and 64 bit
- Ubuntu 18.04, Ubuntu 20.04 64 bit
- CentOS 7 64 bit

## 5.2 Building the binaries

The build system for the software binaries will be initially drawn from the one used in the *FCIDECOMP v1.0.2 source code*, and adapted from there to guarantee support for all the required systems.

## 5.3 Packaging as a Conda package

Packages will be built using Conda, as it provides standardised environments with a large set of pre-compiled packages. From the point of view of Conda, the operating systems listed in the *Supported platforms* paragraph can be considered as two groups of OS: in Conda standardised environment it is enough to build the package for one Linux distribution in order to make it compatible with other Linux distributions. So two conda packages will be released: one for Linux distributions, and one for Windows 10.

These conda packages will install both the FCIDECOMP libraries and its Python bindings. As a blueprint for the conda recipes, the *Conda recipe* for the packaging of FCIDECOMP mantained by Martin Raspaud from the Swedish Meteorological and Hydrological Institute will be used.

The initial aim will be to support HDF5 1.10, and then the use of *conda variants* will be explored to support multiple versions on the same build platform.

Conda packages will be uploaded to EUMETSAT Anaconda repository.

## 5.4 CI/CD pipelines

We will implement simple CI/CD pipelines to compile, build, test and possibly upload the conda packages to the conda repository.

At least two GitLab runners will be implemented, one with a Docker executor on Linux and the other on Windows.

# CHAPTER SIX

# DEPLOYMENT

The following paragraphs outline the strategy to deploy the FCIDECOMP package.

## 6.1 CI/CD pipelines

We will implement simple GitLab CI/CD pipelines to compile, build, test and possibly upload the conda packages to EUMETSAT Anaconda repository.

At least two GitLab runners will be implemented, one with a Docker executor on Linux and the other with a Shell executor on Windows.

# CHAPTER SEVEN

# APPENDIX 1 - TRACEABILITY MATRIX

CHAPTER

EIGHT

# APPENDIX 2 - LIST OF USERS AND DEVELOPERS CURRENTLY USING FCIDECOMP

CHAPTER

NINE

# APPENDIX 3 - DESIGN JUSTIFICATIONS