

Developing a FEWS Adapter based on NetCDF-CF

- [Developing a FEWS Model Adapter \(NetCDF-CF\) \(Since FEWS 2014.01\)](#)
 - [NetCDF Files](#)
 - [Running model adapters within FEWS](#)
 - [NetCDF Viewers](#)
 - [CDL Format](#)
 - [Model and Pre- and Post-adapters.](#)
 - [NetCDF-Run-File.](#)
 - [Logging.](#)
 - [Placeholder Files](#)
 - [Model States \(restart files\).](#)
 - [Location and variable id mapping](#)



This is an overview of some of the requirements for developing a Delft-FEWS compliant adapter. This list is by no means exhaustive so before starting to develop an adapter please contact fews.support@deltares.nl

Developing a FEWS Model Adapter (NetCDF-CF) (Since FEWS 2014.01)

A model adapter is the interface between a so-called model and the Delft-FEWS system. It enables FEWS to run such a model, thus providing the essential forecasting functionality. For each particular FEWS application applying a model, however, some aspects of this system have to be configured by the user in order for the system to work correctly. To achieve this, it is useful that the user has at least some basic understanding of the relation between Delft-FEWS, the model adapter and the forecasting model. In this section, a brief overview of this relation is provided. For more information, the user is referred to the [FEWS manual: General Adapter..](#)

Also make sure to use the following new configuration options that have been added to the FEWS General Adapter recently. Use these options for handling log files and state files and writing placeholder files, so that model adapters do not need to do those things themselves:

- [General Adapter Module - exporting model state files without using PI state description files](#)
- [General Adapter Module - importing model state files without using PI state description files](#)
- [General Adapter Module - parsing model \(adapter\) log files without using PI diagnostic files](#)
- [General Adapter Module - exporting placeholder NetCDF files that can be filled with data by a model post adapter](#)

NetCDF Files

Since FEWS 2014.01 it is possible to use only NetCDF files (no-xml) for the communication between FEWS and the model adapter. Good libraries for reading and writing NetCDF files exist for several languages. XML files are a lot more complex to read and to write.

- Java: <http://www.unidata.ucar.edu/software/thredds/current/netcdf-java/documentation.htm>
- Fortran: <https://www.unidata.ucar.edu/software/netcdf/docs/netcdf-f90/>
- python, <http://unidata.github.io/netcdf4-python/>
- R-project <http://cran.r-project.org/web/packages/RNetCDF/index.html>
- C: <http://www.unidata.ucar.edu/software/netcdf/docs/netcdf-c/>
- C++: <http://www.unidata.ucar.edu/software/netcdf/docs/netcdf-cxx/>

NetCDF time series and attributes can be read with just a few lines of source code.

For more information about the NetCDF format in general, see [NetCDF formats that can be imported in Delft-FEWS](#) and the links on that page.

Running model adapters within FEWS

If you create a model adapter in one of these languages, you can use the model adapter within FEWS in the following way:

- Java: Copy the *.jar files somewhere within your configuration (e.g. %REGION_HOME%\Modules\bin\<model_name>_adapter/. Refer to this location in the General Adapter when you want to execute (part of) the model adapter in an <executeActivity>.
- Fortran:
- Python: [Python model adapters within FEWS](#)
- R-project:
- C:
- C++:

NetCDF Viewers

To inspect the NetCDF files there are several viewers available.

They are all free and all have their strengths

Panoply : <http://www.giss.nasa.gov/tools/panoply/>

ncBrowse: <http://www.epic.noaa.gov/java/ncBrowse>

intel array visualizer: <http://software.intel.com/en-us/articles/intel-array-visualizer>

CDL Format

For NetCDF there is also a text representation available. <http://www.unidata.ucar.edu/software/netcdf/docs/netcdf/CDL-Syntax.html>
 This is very useful for debugging, documentation and as input/output for unit tests
 Converting from NetCDF to CDL
 ncdump: <https://www.unidata.ucar.edu/software/netcdf/docs/netcdf/ncdump.html>
 Converting from CDL to NetCDF
 ncgen: <http://www.unidata.ucar.edu/software/netcdf/docs/netcdf/ncgen.html>

Model and Pre- and Post-adapters.

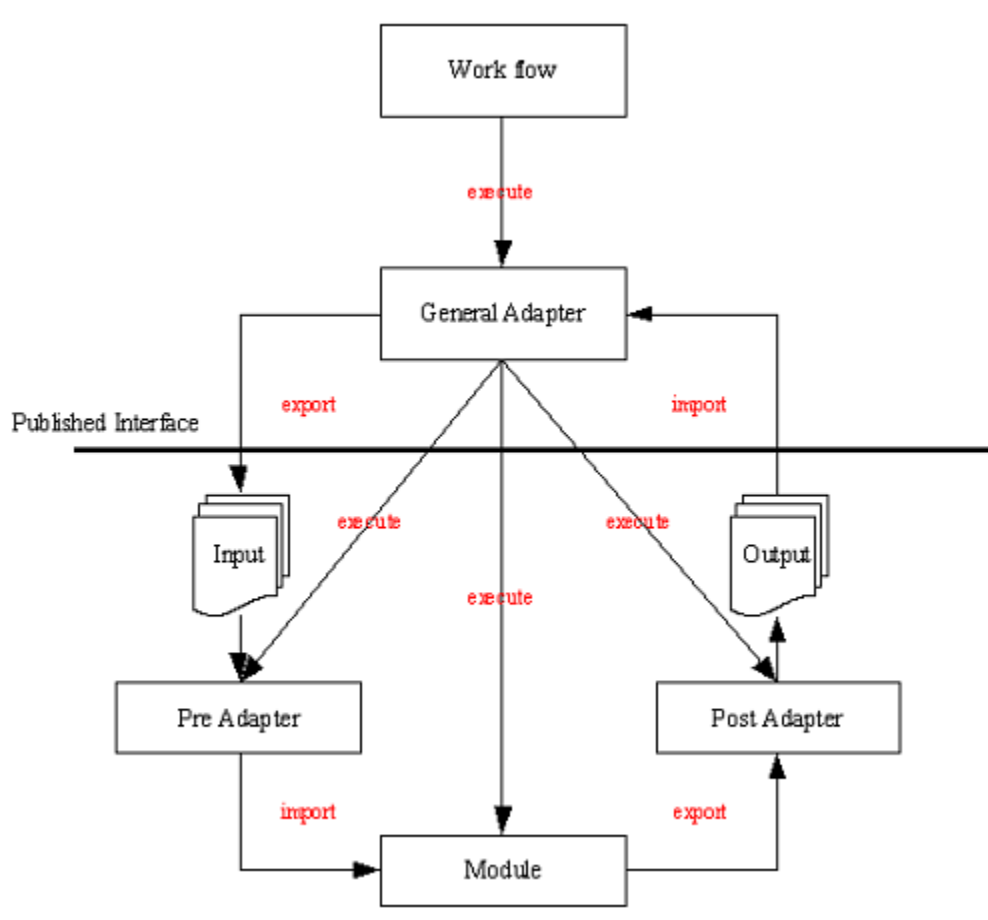


Figure 1: schematic interaction between the FEWS and an external module

A pre-adapter is converting the NetCDF files exported by FEWS to input files for the model.

A pre- adapter is also adjusting the start and end times in the model configuration files so the model will run the right number of time steps .

The post-adapter is converting the output of the model to NetCDF files that are FEWS compliant and can be imported by FEWS.

Not required, but recommended to use a pre- and post-adapter. It is also possible to wrap the model. This wrapper is a pre- and post-adapter in one and it also starts the model and waits for the simulation to finish.

NetCDF-Run-File.

The NetCDF run file contains metadata about the run that has to be executed. The run file is every time re-generated by FEWS before the adapters and models are executed. The most important information is the start and end time of the simulation. The run file can contain extra configuration options as properties. The properties are key-value pairs that are configured in FEWS and added to the run file when the file is generated by FEWS.

Example of the PI run file by ncdump.

```

netcdf run_info {
dimensions:
    path_length = 255 ;
    log_level_length = 5;
    input_netcdf_file_count = 2 ;
    input_state_file_count = 1 ;
    output_netcdf_file_count = 2 ;
variables:
    char log_level(log_level_length) ;
    double start_time ;
        start_time:long_name = "start_time" ;
        start_time:standard_name = "time" ;
        start_time:units = "minutes since 2002-11-26 00:00:00.0 +0000" ;
    double end_time ;
        end_time:long_name = "end_time" ;
        end_time:standard_name = "time" ;
        end_time:units = "minutes since 2002-11-26 00:00:00.0 +0000" ;
    char work_dir(path_length) ;
    char input_netcdf_files(input_netcdf_file_count, path_length) ;
    char input_state_files(input_state_file_count, path_length) ;
    char output_netcdf_files(output_netcdf_file_count, path_length) ;
    char properties ;
        properties:string_property = "zs_0 with spaces" ;
        properties:INT_property = 3 ;
        properties:float_PROPERTY = 3.5f ;
        properties:DOUBLE_PROPERTY_321 = 0.123456789 ;
        properties:logical_property_1 = "true" ;
        properties:logical_property_2 = "false" ;

// global attributes:
    :title = "Run file" ;
    :institution = "Deltares" ;
    :source = "Run file from Delft-FEWS" ;
    :history = "2014-03-18 12:22:55 GMT: exported from Delft-FEWS to D:
\\fews\\workspace\\fews\\junit_test_output\\nl\\wldelft\\fews\\system\\plugin\\generaladapter\\exportDir\\run_in
fo.nc" ;
    :references = "http://www.delft-fews.com" ;
data:
    logLevel = "debug";

    start_time = 0 ;

    end_time = 5880 ;

    work_dir = "..\\work" ;

    input_netcdf_files =
        "..\\work\\boundary_data.nc",
        "..\\work\\more_boundary_data.nc" ;

    input_state_files =
        "..\\input_state\\state.inp" ;

    output_netcdf_files =
        "..\\work\\model_output_data1.nc",
        "..\\work\\model_output_data2.nc" ;

    properties = " " ;
}

```

Logging.

Adapters and models should write one or more log files. It is easiest to let them be plain text files. Log message should be recognizable as debug, info, warn or error by a pattern. In FEWS this pattern (for example "ERROR:") is configured. FEWS can read those lines and map them to a log level of choice: [GeneralAdapterModule-logFile](#)

When FEWS finds an error log line the simulation will be marked as failed.

Placeholder Files

FEWS can create empty output time series so called placeholder files. The placeholder files are CF compliant and contains all the variables, dimension and locations FEWS expects but with zero time steps. The post adapter can easily add the time steps without bothering about CF compliance. As an optimization the pre-adapter can adjust the model configuration so the model only creates output for the locations and variables FEWS wants to import.

Model States (restart files).

A model state makes it possible to continue a run later. The state files that are exported by FEWS and will be imported after the simulation are listed in the run. When the model state contains time information it is sometimes desired that the pre-adapter adjust this time to allow the states to be used for a different time to reduce the warming up period of a simulation. The pre-adapter can adjust the model configuration to switch on/off the generation of a state file at the end of a run. The pre-adapter knows if states are imported after the simulation by querying the "output_netcdf_files" variable in the run file.

Location and variable id mapping

The pre-adapter and post adapter don't have to bother about translating the FEWS location ids and parameter ids to model location and variable ids. This mapping can be configured in FEWS. When the CF standard names are configured for the FEWS parameters the standard names will also be written to the exported NetCDF files and recognized while importing. FEWS is also exporting the location coordinates to the NetCDF input and placeholder files. Some adapters choose to use the coordinates and/or the standard names for the mapping.