Delft3D adapter with NetCDF run file

This page briefly describes the Delft3D model adapter for Delft-FEWS when using a **NetCDF run file** as input. For how to use a delft3dModel adapter xml file as input, please refer to the page Delft3D adapter.

The new version of the Delft3D model adapter can use a run info file in NetCDF format as input. The old way of using a delft3dModel adapter xml file as input is also still supported (see Delft3D adapter). When using a NetCDF run info file, the delft3dModel adapter xml file is not needed anymore. In this case the information that used to be in the delft3dModel adapter xml file is read from the NetCDF run file, which is generated by Delft-FEWS using the configuration in the FEWS general adapter config file. This makes the configuration easier, since all information to run the Delft3D model adapter is now present in the FEWS general adapter file that is used to run the Delft3D model (single point of configuration).

This model adapter presently supports the following modules from the Delft3D suite:

- Delft3D-FLOW
- Delft3D-WAQ (Delwaq)
- Delft3D-WAVE
- Delft3D-PART
- D-Flow FM

In order to use the adapter some knowledge of both Delft-FEWS and Delft3D is required.

Directory structure

To minimise the number of choices that have to be made by the user, the Delft3D adapter expects a fixed set of directories and files. Roughly speaking: anything for which a reasonable default can be set, has been fixed. This means for example that the description of the export activities in the Delft-FEWS configuration files must use these directory and file names. The advantage is that there are much less opportunities for making mistakes (see the section on configuration errors).

The following directories and files are used, even if they are not required for the simulation. The directories are taken as *subdirectories* of the <rootDir> that you give in the configuration of the general adapter module. Some of these directories and files are only needed if state handling in the pre/post adapter is switched on.

Directory/file	Purpose
input	Contains all the files with timeseries and map stacks exported by Delft-FEWS
input/timeseries.xml	XML-file with (scalar) timeseries
input/map_<param>.xml	XML-file describing the map stacks with parameter "param" (see the documentation of the keywords)
stateInput	Contains all the files with the initial conditions and other static information that together constitute the "state" from which the computation must start. This directory is only needed if state handling is switched on.
stateInput/export_states.xml	The XML-file describing the time of the state files (Export from the point of view of Delft-FEWS). This file is only needed if state handling is switched on.
output/timeseries_<runId>.xml	XML-file with the resulting (scalar) timeseries, to be imported by Delft-FEWS
output/fewsParameter>.xml	XML-file describing the resulting map stacks with FEWS parameter "FEWS-param" (see the documentation of the keywords)

stateOutput	Contains all the files with the final results, useful as initial conditions. This directory is only needed if state handling is switched on.
stateOutput /import_states.xml	The XML-file describing the time of the final state files (Import from the point of view of Delft-FEWS). This file is only needed if state handling is switched on.

NetCDF run file

The Delft3D adapter uses a fixed set of directories and files to do most of its work. Some items do have to be specified as properties in the NetCDF run file (to be placed in the root of the <rootDir>), as these may vary from application to application. This section describes the properties that are needed.

Properties

module	(optional)	String indicating the type of Delft3D module to use. Can be "FLOW", "FLOW_FM", "WAQ", "ECO", "WAVE" or "PART". The module can also be specified as a runtime argument for the pre and post adapter, in which case it is not required in the netcdf run file. If the module is both present as a runtime argument and as a property in the netcdf run file, then the runtime argument overrules the property in the netcdf run file.
run_id	(required)	String used to identify the template input file. The runId is used by the pre-adapter to determine which input files to scan for placeholder keywords. *Relevant input files for each Delft3D module are listed below.*
template_directory	(required)	Template directory relative to rootDir. Directory containing the template files. These files are copied to the workDir by the model adapter prior to running a simulation. For each module only the files that correspond to that module and can contain keywords are copied from the template directory to the work directory. All other files (i.e. files that are not template files at all) should be present in the work directory in advance, as these files will not be copied over.
state_file_id	(optional)	Prefix of the state file filename. This is only used for WAVE pre-adapter and WAVE post-adapter. The WAVE pre-adapter only processes state files that start with this prefix. This option is only used if state handling is switched on.
auxiliary_grid_file	(optional)	Both Delft3D-FLOW and Delft3D-WAQ require the name of a grid file: • Delft3D-FLOW requires the name of the file (*.grd) that defines the grid for the meteorological data. It is written in the header of the file with air pressure and so on. • Delft3D-WAQ requires the name of the LGRID and CCO files, so that the segment function files can be written properly. This information is needed by the post-adapter as well, in order to export the results on a grid.
field_file_format	(optional)	Can be "old" (same behaviour as when not configured), "new", "CURVILINEAR" (same behaviour as for "new"). Control the format of the meteo and other field files.
output_time_series_in_one_file	(optional)	Can be "true" or "false", default is false. Merges all output timeseries to one output XML file if true.
online_morphology	(optional)	Can be "true" or "false", default is false. If true, then all *.dep files (morphology files) in the stateInput folder and subfolders thereof are copied to the workDir by the pre-adapter. This option is only used if state handling is switched on.
restart_netcdf_file_subscript	(only required for module FLOW_FM)	The input restart file for Delft3D always has to be called "tri-rst.<run_id>.rst" where <run_id> is the value of the run_id property. The input restart file for FLOW_FM always has to be called "tri-rst. <run_id><restart_netcdf_file_subscript>" where <run_id> is the value of the run_id property and <restart_netcdf_file_subscript> is the value of the restart_netcdf_file_subscript property. Usually restart_netcdf_file_subscript = "_rst.nc". This option is only used if state handling is switched on.
initial_conditions_id	(optional)	Can be null. This is only used for FLOW pre-adapter and WAQ pre-adapter. This is ignored if there are multiple state files (from other modules) for online-coupled modules and/or domain decomposition. This option is only used if state handling is switched on.
invert_time_stamp	(optional)	Can be "true" or "false", default is false.
input_spectra_file_to_convert	(optional)	Pathname of netcdf file with input spectra data that should be converted. This should be either an absolute path or a path relative to the workDir specified in the netcdf run file. This only works if the module property is set to "WAVE". Multiple spectra files can be specified by seperating them with a semicolon, only use a semicolon between spectra files and not at the end. For example: <string key=" input_spectra_file_to_convert" value="%ROOT_DIR%/workDir/AGM_01.nc; %ROOT_DIR%/workDir /AGM_02.nc; %ROOT_DIR%/workDir/AGM_03.nc"/>
overruling_geodatum	(optional)	Overrules the default (WGS84) geodatum for writing and reading spectra locations, used by the Delft3D-WAVE adapter. For example: <string key="overruling_geodatum" value="Rijks Driehoekstelsel"/>

Module	Template files	Purpose
FLOW	<runid>.mdf <runid>.bcc, .bct, . dis, ...	Master definition file. Various attribute files.
WAQ, ECO	<runid>.inp couplnef.inp	Main input file, the only file that is supposed to contain keywords for this module. Input file for the coupling program.
PART	<runid>.inp	Like WAQ.
WAVE	<runid>.mdw	
FLOW_FM	<runid>.mdu	see MD - DFLOWFM model adapter

Notes for users

- Currently the adapter assumes that the model always runs in time zone GMT (this will be improved in future).
- The same NetCDF run file can be used for both the pre-adapter and the post-adapter.
- The startTime and endTime from the NetCDF run file (which are determined by FEWS) are used as the time frame of the simulation.
- The modelDir option from the old delft3dModel adapter config file has been replaced with the property "template_directory". For each module only the files that correspond to that module and can contain keywords are copied from the template directory to the work directory. All other files (i.e. files that are not template files at all) should be present in the work directory in advance, as these files will not be copied over.
- The option useWaqMapFilesForInitialConditions from the old delft3dModel adapter config file is now assumed to be always true and cannot be configured, i.e. the new format is always used.
- The option to specify the "module" property as a runtime argument (see above) can be used to run the pre or post adapter multiple times with the same netcdf run info file but for different modules. For instance for a coupled FLOW-WAVE Delft3D run, need to configure five execute activities in sequence:
    1. pre adapter for module FLOW.
    2. pre adapter for module WAVE.
    3. FLOW-WAVE model.
    4. post adapter for module FLOW.
    5. post adapter for module WAVE.
- The state handling in the pre and post adapter is automatically switched on if the input state description file stateInput/export_states.xml exists. If this file does not exist, then the state handling in the pre and post adapter is automatically switched off. Switching off the state handling in the pre and post adapter makes the configuration much easier. For more information, see 05 General Adapter Module#05GeneralAdapterModule-exportStateActivity and 05 General Adapter Module#05GeneralAdapterModule-importStateActivity

Commandline options

The first runtime argument for the pre or post adapter must be the root directory.

The second runtime argument must be the path and name of the netcdf run info file, relative to the root directory.

The third runtime argument is optional and can be a string indicating the type of Delft3D module to use (also see property "module" above). This can be "FLOW", "FLOW_FM", "WAQ", "ECO", "WAVE" or "PART". If the module is both present as a runtime argument and as a property in the netcdf run file, then the runtime argument overrules the property in the netcdf run file.

-swapNorthSouth

Since 2013.02 there is a command line option to swap north and south in the Delft3DPostAdapter for regular grids.

```
<executeActivity>
        <description>Delft3D Adapter</description>
        <command>
                <className>nl.wldelft.fews.adapter.delft3d.Delft3DPostAdapter</className>
        </command>
        <arguments>
                <argument>%ROOT_DIR%</argument>
                <argument>run_info.nc</argument>
                <argument><module (optional)></argument>
                <argument>-swapNorthSouth</argument>
        </arguments>
        <timeOut>10800000</timeOut>
        <overrulingDiagnosticFile>%ROOT_DIR%/diagnostics/delft3dpostadapter.xml</overrulingDiagnosticFile>
</executeActivity>
```

-skipFirstTimeStep

Since 2013.02 there is a command line option to skip the first timestep of a forecast in the Delft3DPreWaveAdapter. This is to be used for forecasts and hindcasts so that the warm state will be picked up correctly due to inconsistencies in the way state files are written between FLOW and WAVE. From Delft3D version 6.02.01.5425 (Sep 2015) and newer this is no longer required. For models using **online coupled FLOW and WAVE** it is highly recommended to use Delft3D version 6.02.01.5425 or higher **without** the -skipFirstTimeStep option.

Import types

The pre-adapter converts the netcdf file specified in the optional property "input_spectra_file_to_convert" to a file in SWAN ascii spectra format with the same path and name but different extension (.BND). For example the original file input/spectra.nc is converted to input/spectra.BND. If the property "input_spectra_file_to_convert" is not specified, then this step does nothing.

Export types

The post-adapter converts all output from the model output files from the model format to the FEWS pi format. Scalar timeseries are written in the XML file timeseries_<runId>.xml. 2D maps are written in ArcInfo ASCII files called <fewsParameter>.xml. Currently only output for layer=1 is converted (this may be improved in future). Note that the layer will show up as qualifierId in pi time series xml files. For grid data the locationId is set to "unknown", i.e. fews external locationId should also be "unknown". The names of the files to import into FEWS and the external location/parameter ids in the FEWS idMapping should equal the model ids, i.e. the ids used by the Delft3D model.

Furthermore, if the module property is set to "WAVE", then the post-adapter converts all model output .sp2 files in the workDir from the model format to the NetCDF format and merges them into a single NetCDF file called "wave_spectra_output.nc" in the workDir (only if .sp2 files are present in the workDir). So this takes care of converting the output from module WAVE, if present.

Template files and keywords

In the table below the various keywords are explained. It is important to note that these keywords are filled whenever they are encountered in a template file – the keywords are not associated with a particular type of file. The format for the keywords and any arguments there may be is as follows:

$(*keyword: argument1, argument2, ...*)

or, if there are no arguments:

$(*keyword*)

(Technical note: the entire text from the opening "$(" to the trailing ")" is replaced, without a trailing new line, by whatever the contents should be. The arguments should not contain commas parenthesis and there should be at least one space after the comma. This syntax makes it easy to implement the template mechanism.)

Keyword	Description
FLOW_TIME_START	Start of the simulation (format in accordance with Delft3D-FLOW). (This is actually the time in minutes since the reference time, found in the mdf-file)
FLOW_TIME_STOP	Stop of the simulation (format in accordance with Delft3D-FLOW) (Ditto as the start time)
FLOW_TIME_RST	Duration of the simulation – useful for setting the time interval of writing the restart files so that only one restart file is written at the end of the simulation.
FLOW_TIMESERIES	Placeholder to fill in the timeseries in the so-called tim format (only the time and data, not the header). The keyword should be followed by the names of all timeseries that should be filled in there, separated by a comma and one or more spaces: $(FLOW_TIMESERIES: h/bound-1 salinity/bound-1, temperature/bound-1) (Despite the fact that the "tim" format is used more widely than just FLOW, there are some specifics involved.)
FLOW_MAPSTACK	Placeholder for the name of the file that will hold the scalar field data (as found in the map stack files). It should be followed by the name of the parameter: $(FLOW_MAPSTACK: pressure) Then: • The string "FLOW_MAPSTACK pressure" is replaced by "pressure.dat" • The file "pressure.dat" contains the map stacks in the format proper to Delft3D-FLOW • The XML-file "map_pressure.xml" in the input directory contains the details: which ASCII files and so on.
WAQ_TIME_START	Start of the simulation (format in accordance with Delft3D-WAQ/ECO: yyyy/mm/dd-hh:mm:ss)
WAQ_TIME_STOP	Stop of the simulation (format in accordance with Delft3D-WAQ/ECO: yyyy/mm/dd-hh:mm:ss)
WAQ_TIMESERIES	Placeholder to fill in the timeseries in the WAQ /ECO format (only the time and data, not the header). The keyword should be followed by the names of all timeseries that should be filled in there, separated by spaces: $(WAQ_TIMESERIES: salinity/bound-1, temperature/bound-1)

WAQ_MAPSTACK	Placeholder for the name of the file that will hold the scalar field data (as found in the map stack files). It should be followed by the name of the parameter: $(WAQ_MAPSTACK: windvel) Then: • The string "WAQ_MAPSTACK windvel" is replaced by "windvel.dat" • The file "windvel.dat" contains the map stacks in the format proper to Delft3D-WAQ/ECO • The XML-file "map_windvel.xml" in the input directory contains the details: which ASCII files and so on.
PART_RUNID	The run ID for the Delft3D-PART computation (used in the filename.dat file)
PART_TIMESERIES	Placeholder to fill in the timeseries in the PART format, similar to the WAQ and FLOW timeseries keywords.
COUP_TIME_START	Start time of coupling (in seconds; time frame is that of Delft3D-FLOW)
COUP_TIME_STOP	Stop time of coupling (in seconds; ditto)
P.M.	Keywords specific to Delft3D-WAVE

For examples, see the MDF, MDW, BCC, WND, BND and INP files attached to the WiKi page Delft3D adapter.

Example configuration generalAdapterRun

Execute activites

The Delft3D model adapter and executable can be executed from FEWS in the following way:

```
<executeActivities>
   <executeActivity>
      <description>Delft3D Adapter</description>
      <command>
         <className>nl.wldelft.fews.adapter.delft3d.Delft3DPreAdapter</className>
      </command>
      <arguments>
         <argument>%ROOT_DIR%</argument>
         <argument><netcdf run file></argument>
         <argument><module (optional)></argument>
      </arguments>
      <timeOut>10800000</timeOut>
      <overrulingDiagnosticFile>%ROOT_DIR%\diagnostics\delft3dpreadapter.xml</overrulingDiagnosticFile>
   </executeActivity>
   <executeActivity>
      <description>Run delftflow</description>
      <command>
         <executable>bin/deltares_hydro.exe</executable>
      </command>
      <arguments>
         <argument>config-flow2d3d.ini</argument>
      </arguments>
      <timeOut>90000000</timeOut>
      <overrulingDiagnosticFile>%ROOT_DIR%/dcsmv5_Diagnostic_Placeholder.xml</overrulingDiagnosticFile>
   </executeActivity>
   <executeActivity>
      <description>Delft3D Adapter</description>
      <command>
         <className>nl.wldelft.fews.adapter.delft3d.Delft3DPostAdapter</className>
      </command>
      <arguments>
         <argument>%ROOT_DIR%</argument>
         <argument><netcdf run file></argument>
         <argument><module (optional)></argument>
      </arguments>
      <timeOut>10800000</timeOut>
      <overrulingDiagnosticFile>%ROOT_DIR%\diagnostics\delft3dpostadapter.xml</overrulingDiagnosticFile>
   </executeActivity>
</executeActivities>
```

The Delft3D model executable can be called directly (as shown in the above example), or by calling an external batch file to execute the model.

For examples, see the generalAdapter_example_*.xml files attached to the WiKi page Delft3D adapter.

Configuration errors

The adapter checks for the following types of errors:

- Are all the timeseries and map stacks as encountered in the template files indeed available from the export by Delft-FEWS? If not, the proper data can not be filled in in the files and the computation can not run properly.
- Do all timeseries filled in in a particular table have the same start and stop time and the same time step? (The adapter does not attempt to correct any mismatch – this should be done via the Delft-FEWS configuration)

HowTo's

Grid definition Delft3D in FEWS

To import gridded output from Delft3D into FEWS, a grid definition needs to be specified in the FEWS grids.xml file. This grid file can be generated using the Matlab script print_Delft3D_grid.m attached to the WiKi page Delft3D adapter.

State management

FEWS required to have a cold state file for each Delft3D model in the ColdStates folder (zipped). This cold state file is a Delft3D restart file which needs to be generated by the model developer up-front (outside of FEWS), and needs to describe representative initial conditions.

The below configuration shows an example of the exportStateActivity to export a Delft3D model state from the FEWS generalAdapter.

```
<exportStateActivity>
   <moduleInstanceId>DCSM_Historical</moduleInstanceId>
   <stateExportDir>%ROOT_DIR%/stateInput</stateExportDir>
   <stateConfigFile>%ROOT_DIR%/stateInput/export_states.xml</stateConfigFile>
   <stateLocations type="file">
      <stateLocation>
         <readLocation>dcsm98.res</readLocation>
         <writeLocation>dcsm98.res</writeLocation>
      </stateLocation>
   </stateLocations>
   <stateSelection>
      <warmState>
         <stateSearchPeriod unit="hour" start="-48" end="-1"/>
      </warmState>
   </stateSelection>
</exportStateActivity>
```

From the Delft3D-FLOW file this initial state file is subsequently called in the following way:

```
Commnt=
Restid= #dcsm98.rst#
Commnt=
```

The warmState search period should be sufficiently long to spin-up the model in case of a cold start.

The use of pi state description files is optional for both pre and post adapter. The example below shows a configuration of an exportStateActivity without using pi state description files (i.e. much simpler from a user's perspective):

```
...
        <timeZoneName>GMT+0:00</timeZoneName>
    </timeZone>
    <endDateTimeFormat>yyyyMMdd.HHmmss</endDateTimeFormat>
</general>
...
<exportStateActivity>
    <moduleInstanceId>$MODULE_INSTANCE_ID$</moduleInstanceId>
    <stateExportDir>%WORK_DIR%</stateExportDir>
    <stateSelection>
        <warmState>
            <stateSearchPeriod unit="hour" start="-6" end="-1"/>
        </warmState>
    </stateSelection>
</exportStateActivity>
...
<importStateActivity>
    <stateFile>
        <importFile>%WORK_DIR%/tri-rst.dcsm5.res</importFile>
        <relativeExportFile>tri-rst.dcsm5.rst</relativeExportFile>
    </stateFile>
    <stateFile>
        <importFile>%WORK_DIR%/hot_1_%END_DATE_TIME%.res</importFile>
        <relativeExportFile>hot_1_00000000.000000</relativeExportFile>
    </stateFile>
    <expiryTime unit="day" multiplier="3"/>
</importStateActivity>
...
```