Telemac

Table of Contents

TELEMAC 2D MODEL ADAPTER

TELEMAC 2D

The TELEMAC-2D is an open source code (http://www.opentelemac.org/). It solves depth-averaged free surface flow equations as derived first by Barré de Saint-Venant in 1871. The main results at each node of the computational mesh are the depth of water and the depth-averaged velocity components. The main application of TELEMAC-2D is in free-surface maritime or river hydraulics.

TELEMAC 2D Model Adapter – Summary

This page describes the TELEMAC 2D module adapter and its functions.

In general, the TELEMAC 2D Model Adapter will use a steering properties file, in NETCDF format, to run the model and carry out Pre and Post processes.

The Pre Adapter transforms Delft FEWS exported data, in NetCDF format, into the model specific format. This data includes space and time varying wind fields and hydrodynamic boundary conditions. Additionally, it adapts a template configuration file (*.cas file) into a working configuration file.

The Run model executes the Telemac model. The routines in this module are created specifically to run the model remotely in an IMDC cluster, additionally, it is possible to run the model locally in a personal computer (NOT TESTED).

The Post Adapter main aim is to transform the model results from their native format into NetCDF format which can be read by the Delft FEWS. The map files are transformed into the new developed UGRID-0.9 netcdf convention for unstructured grids (CF-1.6 UGRID-0.9). There are curently 4 output variable configures in the code (VELOCITY U, VELOCITY V, WATER DEPTH, FREE SURFACE) but any Telemac variable can be exported.

Note from the developer: The Model adapter is still in testing phase, the adapter is still not bug free!!!

System Requirements

The TELEMAC model adapter was developed using Python language, therefore it requires:

- Python 2.7.5 with 32 Bits
- Sys Python Package
- Numpy Python Package
- Pylab Python Package
- Scypy Python Package
- Datetime Python Package
- ParserSELAFIN Python scripts developed by Christopher J. Cawthorn and Sebastien E. Bourban (HR Wallingford and EDF) https://github.com/ogoe/OpenTelemac-svn-mirror/blob/master/pytel/parsers/parserSELAFIN.py
- netCDF4 Python Package by Unidata - http://netcdf4-python.googlecode.com/svn/trunk/docs/netCDF4-module.html
- Paramiko Python package for cluster operations

The developers recommend the installation of the 32 bits free version of Anaconda Scientific Python Distribution by Continuum Analytics (https://store.continuum.io/cshop/anaconda/), with Python 2.7.5. The full installation mode will mount into the computer mostly all of the packages. After installation use the built in installation tool (conda install) to download and install netCDF4 Python package and paramiko Python Package.

The steering properties file

This file is a NETCDF file created by Delft FEWS. It is configured using the exportRunFileActivity in the General Adapter and through it is configured and passed the properties to the model adapter. The steering file must have this properties:

Properties	Description
castmp*	Path name of template of Telemac Configuration File (*.cas)
caswrk*	Path and Name of the Working Telemac Configuration File
geoslfdyn	Name and Path of the geometry file with extra dynamic information (eg. Wind) in Selafin format. NOTE: If the geometry file is static this Property is not required
geoslftmp	Path of Selafin template of the Geometry file (eg. Geometry file itself) with which the Adapter will fill with the info (NOTE: only required if geoslfdyn is present)
geoncinf	Path of the NETCDF with information (exported by Delft FEWS) which will be added to the slf template (NOTE: only required if geoslfdyn is present)
geodyntype	Type of information (EG. WIND) that will be added in the dynamic geometry file. (NOTE: only required if geoslfdyn is present). Currently the developers have only implemented the code to read meteorological information (wind and pressure) from the NETCDF and add it to the selafin geometry file (for more details see source code of pre Adapter). Additionally, the modeler has to configure the Fortran File so the model can read information from the geometry file.
liqbc**	Name (as written in the cas file) and Path where the Liquid boundary file is read by Telemac
liqbctmp	Name and Path of the template of the Liquid boundary file (NOTE: only required if liqbc is present)
liqbcnc	Name and Path of NETCDF with information (exported by FEWS) that will be added to the liquid boundary (NOTE: only required if liqbc is present)
fordatafile**	Name and Path of the user defined formatted data file
fordatanc	Path of the NETCDF with information (exported by Delft FEWS) which will be added to the user defined formatted data file. NOTE: STILL IN DEVELOPMENT
fordatatype	Type of formatted data type. Currently available “imdc-ascii” type.
statein	Name and Path of the hotstart. This location is where the DELFT FEWS will export the hotstart from its database.
hotstart	Name and Path of the hotstart that will be read by Telemac
stateout	Path of the hotstart file to be retrieve by FEWS
resultsslf *	Path and Name of the results file
resultstsnc**	Path and Name of the results file in NETCDF format. The information of this file will be used for importing time series in specific locations. The DELFT FEWS needs to provide an empty NETCDF with just headers. This can be configured in the importNetcdfActivity, setting exportPlaceholderFile as TRUE.

resultsfieldnc**	Path of the results file transform into NETCDF format (from here it will be imported back to FEWS). This file will possess the results as fields. Currently the data has to be treated as scalar (instead of grid) in the timeSeriesSets under importNetcdfActivity. Unstructured Grids is still not supported by DELFT FEWS.
The properties with * are required properties. ** means at least one of this properties is required. The rest of properties are optional.	

This is an example of the NETCDF steering properties file:

```
root group (NETCDF3_CLASSIC data model, file format NETCDF3):

title: Run file

institution: Deltares

source: Run file from Delft-FEWS

history: 2014-10-07 08:16:37 GMT: exported from Delft-FEWS to Z:
\projects\OFS\Fews\test_NestingZeebrugge\testrunFEWS\Telemac\forecast\Port_Zeebrugge\run.nc

references: http://www.delft-fews.com

dimensions(sizes): log_level_char_length(5), path_length(255), input_netcdf_file_count(1), output_netcdf_file_count(1)

variables(dimensions): float64 start_time(), float64 end_time(), |S1 log_level(log_level_char_length), |S1 work_dir(path_length), |S1 input_netcdf_files(input_netcdf_file_count,path_length), |S1 output_netcdf_files(output_netcdf_file_count,path_length), |S1 properties()

groups:
```

And properties variable inside the NETCDF:

```
<type 'netCDF4.Variable'>
|S1 properties()
castmp: Z:\projects\OFS\Fews\test_NestingZeebrugge\testrunFEWS\Telemac\forecast\Port_Zeebrugge/templates/temp_cas_HD64.txt
caswrk: Z:\projects\OFS\Fews\test_NestingZeebrugge\testrunFEWS\Telemac\forecast\Port_Zeebrugge/model/cas_Run.txt
liqbc: Z:\projects\OFS\Fews\test_NestingZeebrugge\testrunFEWS\Telemac\forecast\Port_Zeebrugge/model/LQ_BC.txt
liqbctmp: Z:\projects\OFS\Fews\test_NestingZeebrugge\testrunFEWS\Telemac\forecast\Port_Zeebrugge/templates/temp_LQ_BC.txt
liqbcnc: Z:\projects\OFS\Fews\test_NestingZeebrugge\testrunFEWS\Telemac\forecast\Port_Zeebrugge/input/liquid_boundaries.nc
resultsslf: Z:\projects\OFS\Fews\test_NestingZeebrugge\testrunFEWS\Telemac\forecast\Port_Zeebrugge/model/results.slf
resultstsnc: Z:\projects\OFS\Fews\test_NestingZeebrugge\testrunFEWS\Telemac\forecast\Port_Zeebrugge/output/resultsTs.nc
resultsfieldnc: Z:\projects\OFS\Fews\test_NestingZeebrugge\testrunFEWS\Telemac\forecast\Port_Zeebrugge/output/resultsField.nc
unlimited dimensions:
current shape = ()
filling off
```

TELEMAC 2D Pre Model Adapter

To run the Pre model adapter, it is required to setup a small bat file. This file will be executed under the executeActivity element of the General Adapter.

The bat file will call the python interpreter (eg. Python.exe) and will possess two arguments, for example:

c:\<Path of Python Interpreter>\python.exe %1 %2

The arguments, configured under executeActivity of the General Adapter, are:

1. Path and Name of the Telemac Preadaptor Script, called: TelemacAdaptorPre.py
2. Path and Name of the Properties Steering File.

Here is an example of the activity configuration in the General Adapter File:

start up activities

```
<executeActivity>
                <description>Telemac PreAdapter</description>
                <command>
                                <executable>%REGION_HOME%\Modules\bin\Telemac\run_PreAdaptor.bat</executable>
                </command>
                <arguments>
                                <argument>%REGION_HOME%\Modules\bin\Telemac\TelemacAdaptorPre.py</argument>
                                <argument>%ROOT_DIR%\run.nc</argument>
                </arguments>
                <timeOut>10800000</timeOut>
                <ignoreDiagnostics>true</ignoreDiagnostics>
</executeActivity>
```

TELEMAC 2D Run Model Adapter

To run the Pre model adapter, it is required to setup a small bat file. This file will be executed under the executeActivity element of the General Adapter.

The bat file will call the python interpreter (eg. Python.exe) and will depending on the location of the run, cluster or local, will possess a fix amount of arguments. The structure of the bat file follows the same as the Pre Adapter:

c:\<Path of Python Interpreter>\python.exe %1 %2 %3… etc.

The arguments for the Cluster case are:

1. Path and Name of the Telemac Run Adapter Script, called: TelemacAdaptorStart.py
2. Working Directory Folder Path.
3. Name of the working Telemac configuration file *.cas. It must have the same name as the one configured in the **caswrk** property.
4. The tag telling that the run will be carried out remotely, for this case is “cluster”
5. Cluster directory

The arguments for the Local case are:

1. Path and Name of the Telemac Run Adapter Script, called: TelemacAdaptorStart.py
2. Working Directory Folder Path.
3. Name of the working Telemac configuration file *.cas. It must have the same name as the one configured in the **caswrk** property.
4. The tag telling that the run will be carried out locally, for this case is “local”

Here is an example of the activity configuration in the General Adapter File:

start up activities

```
<executeActivity>
                <description>Run Telemac</description>
                <command>
                                <executable>%REGION_HOME%\Modules\bin\Telemac\run_StartAdaptor.bat</executable>
                </command>
                <arguments>
                                <argument>%REGION_HOME%\Modules\bin\Telemac\TelemacAdaptorStart.py</argument>
                                <argument>%ROOT_DIR%/model</argument>
                                <argument>cas_Run.txt</argument>
                                <argument>cluster</argument>
                                <argument>/mnt/data/projects/OFS/Fews/test_NestingZeebrugge
/testrunFEWS/Telemac/hindcast/Telemac_zeebrugge_gfs/model</argument>
                </arguments>
                <timeOut>90000000</timeOut>
                <ignoreDiagnostics>true</ignoreDiagnostics>
</executeActivity>
```

TELEMAC 2D Post Model Adapter

To run the Post model adapter, it is required to setup a small bat file. This file will be executed under the executeActivity element of the General Adapter.

The bat file will call the python interpreter (eg. Python.exe) and will possess two arguments, for example:

c:\<Path of Python Interpreter>\python.exe %1 %2.

The arguments, configured under executeActivity of the General Adapter, are:

1. Path and Name of the Telemac Post adapter Script, called: TelemacAdaptorPost.py
2. Path and Name of the Properties Steering File.

Here is an example of the activity configuration in the General Adapter File:

start up activities

```
<executeActivity>
                <description>Telemac PostAdapter</description>
                <command>
                                <executable>%REGION_HOME%\Modules\bin\Telemac\run_PostAdaptor.bat</executable>
                </command>
                <arguments>
                                <argument>%REGION_HOME%\Modules\bin\Telemac\TelemacAdaptorPost.py</argument>
                                <argument>%ROOT_DIR%\run.nc</argument>
                </arguments>
                <timeOut>10800000</timeOut>
                <ignoreDiagnostics>true</ignoreDiagnostics>
</executeActivity>
```

Attachments

TelemacAdaptor.zip Telemac Adapter Scripts v1.0 (not fully debugged)