User Guide for XIOS testcase

Yushan Wang

October 14, 2019

1 What is the XIOS testcase

The XIOS generic testcase is a set of unit tests that test functionalities of XIOS. It is reserved for XIOS developpers and it is designed to ensure that the XIOS does not introduce errors after each code modification.

2 Orgnization of the testcase

The XIOS generic testcase is located in the `GENERIC_TESTCASE` folder inside the XIOS directory. Inside the `GENERIC_TESTCASE` folder, we have for the moment 5 folders for testing specific XIOS functionalities. Under construction

- `test_function`: test the temporal reduce functionalities (accumulate, maximum, minimum, average, ...)
- `test_scalar_algo`: test the scalar algorithms (extract, reduce, transformation, ...)
- `test_axis_algo`: test the axis algorithms (duplicate, extract, interpolate, inverse, transformation, ...)
- `test_domain_algo`: test the domain algorithms (connectivity, expand, extract, interpolate, reorder, transformation, ...)
- `test_grid_algo`: test the grid algorithms (transformation, ...)

3 How to configure XIOS testcase

3.1 Define the configuration variables

At the moment, we provide 7 configuration variables with default values.

- `UsingServer2`: when set to `true`, the two-level server mode is enabled. Otherwise, the classic one-level server mode is used.
- `RatioServer2`: the ratio of the second level servers. Significant only when two-lever server mode enabled.
- `NumberPoolsServer2`: number of the pools of the second level servers. Significant only when two-lever server mode enabled.
- `NumberClients`: number of client processes.
- `NumberServers`: number of servers.
- `Duration`: duration of the simulation.
- `ATMdomain`: type of the domain used in the simulation.

The default values of all 7 configuration variables can be found and modified in the file `default_param.py` located in the root folder `GENERIC_TESTCASE`.

Besides of the default value, one can also set a temporal value for the configuration variables in the file `user_params.def`. In this file, each variable can have multiple values seperated by a semecolon. For example, `NumberServers=2,4,6`. From such definition, the testcase will generate 3 different configurations and tests will be run for each configuration.

3.2 Define the output file check list

The validation of the testcase is ensured by the comparaison between the newly generated output `NetCDF` files and the prestored reference results. In some cases, one maybe interested in only one or serveral output files. It will not be necessary to compare all output files for this will slow down the whole test execuition. To specify the output files to compare with the reference, one can modify the `checkfile.def` file. If nothing is set in this file, or `all` is set, then all output files will be examinated.

4 How to launch XIOS testcase

For the sake of simplicity, we suppose in the following context that the computing resource is Juliot-Curie from the TGCC computing center[1].

Inside the `GENERIC_TESTCASE` folder, we have a sob submission script `job_irene.sh`. This script will execute all the unit tests in all the 5 test folders. We also prepared

[1] `http://www-hpc.cea.fr/en/complexe/tgcc-JoliotCurie.htm`

job scripts for each test folder. If one want to run unit tests for a specific folder, *e.g.* `test_domain_algo`, it is sufficient to launch the `job_irene.sh` script inside the `test_domain_algo` folder.

5 Output of XIOS testcase

An example of the XIOS testcase output is as follows:

```
-- Configuring done
-- Generating done
Test project /GENERIC_TESTCASE
      Start 1: test_domain_algo_config0
 1/16 Test #1: test_domain_algo_config0 ......... Passed 3.51 sec
      Start 2: test_domain_algo_config1
 2/16 Test #2: test_domain_algo_config1 ......... Passed 3.70 sec
      Start 3: test_domain_algo_config2
 3/16 Test #3: test_domain_algo_config2 ......... Passed 2.68 sec
 ...
      Start 14: test_axis_algo_config1
14/16 Test #14: test_axis_algo_config1 ........... Passed 2.31 sec
      Start 15: test_grid_algo_config0
15/16 Test #15: test_grid_algo_config0 ........... Passed 2.23 sec
      Start 16: test_grid_algo_config1
16/16 Test #16: test_grid_algo_config1 ........... Passed 2.24 sec

100% tests passed, 0 tests failed out of 16

Total Test time (real) = 39.63 sec
Built target report
```

This Ctest style summary report shows the overall execution status of all unit tests. Besides of this console output, we also have `report.html` which can be viewed by any web browser.

We also generate job submission script for each configuration in each tet folder. These configuration specified job scripts allow one to focus on one configuration of the given test.

```
/GENERIC_TESTCASE/test_axis_algo/
irene_job_config_UsingSrv2=true_NbServers=2_ATMdomain=lmdz.sh
```

6 References results of XIOS testcase

If no reference results exist for a given configuration, the output `NetCDF` files along with the setup files will be stored in the configuration folder of the test.

```
GENERIC_TESTCASE/test_axis_algo/
      config_UsingSrv2=true_NbServers=2_ATMdomain=lmdz/
            tmp_reference/
            setup/
            xios_output/
      config_UsingSrv2=true_NbServers=4_ATMdomain=lmdz/
            tmp_reference/
            setup/
            xios_output/
GENERIC_TESTCASE/test_domain_algo/
      ...
```

These results are considered to be temporal. Once the results are confirmed, one can store these results elsewhere for later use. We provide the `copy_to_reference.py` script for storing the confirmed reference results in the `$WORK` directory of the Juliot-Curie supercomputer. Similarly, we can retrieve the reference results from the `$WORK` folder to the test folder by using the `copy_from_reference.py` script.