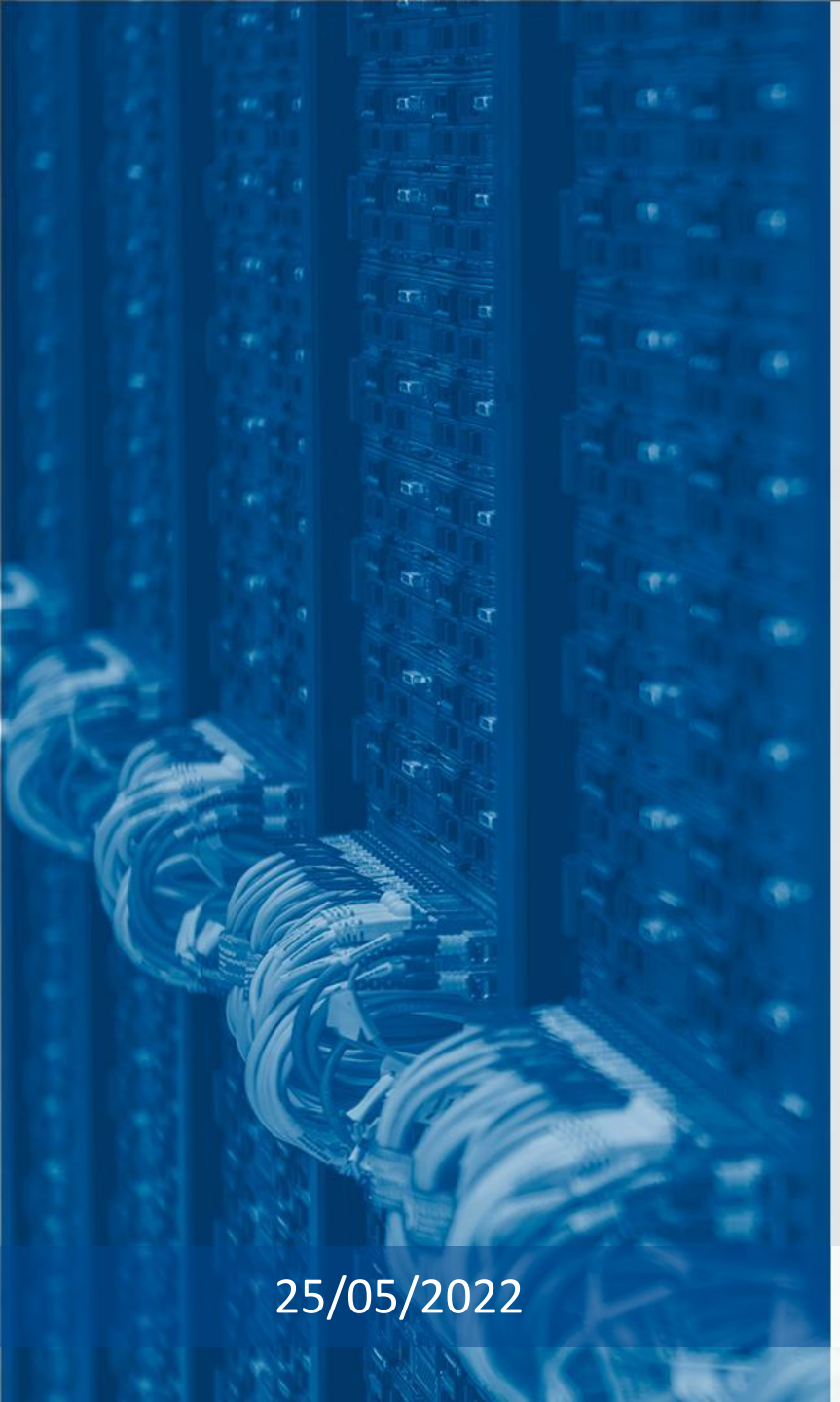


Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Sharing is encouraged

Approach to make an I/O server performance-portable across different platforms: OpenIFS-XIOS integration as a case study

Xavier Yepes-Arbós, BSC
Jan Streffing, AWI
Mario C. Acosta, BSC
Kim Serradell, BSC

esiwace
CENTRE OF EXCELLENCE IN SIMULATION OF WEATHER AND CLIMATE IN EUROPE

The research leading to these results has received funding from the EU H2020 Framework Programme under grant agreement no. 823988
This material reflects only the author's view and the Commission is not responsible for any use that may be made of the information it contains

25/05/2022

EGU General Assembly 2022 , Vienna, Austria

Introduction

- Earth System Models (ESMs) require a large demand of computing power and this might generate a massive volume of model output data that must be **efficiently written** into the storage system.

- The I/O issue is typically addressed by adopting scalable **parallel I/O** solutions:
  - I/O servers -> inline diagnostics.

- I/O servers are complex tools that need to be **tuned** to perform efficiently, that is a trade-off between throughput and resource usage. Tuning scenarios:
  - Different platforms.
  - Different model configurations for a single platform.

Objective

The main objective is to identify and tune a series of important **parameters** that should be considered to make an I/O server **performance-portable** across different platforms.

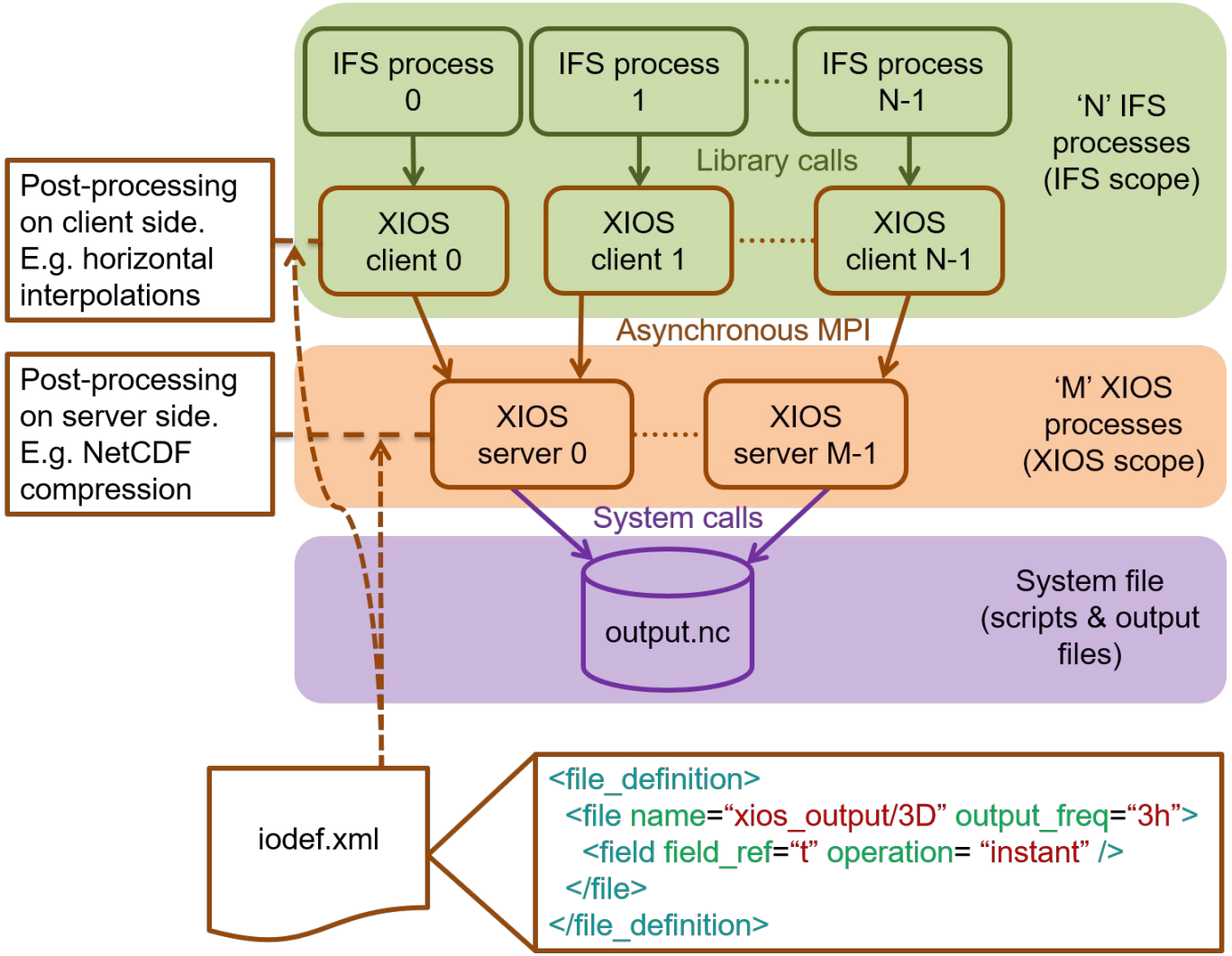
Test case: OpenIFS and XIOS integration

- The XML Input/Output Server (**XIOS**) is an asynchronous MPI parallel I/O server developed by the Institute Pierre Simon Laplace (IPSL). It is a widely I/O tool used in the European ESM community.

- **OpenIFS** is an atmospheric general circulation model developed and maintained by the European Centre for Medium-Range Weather Forecasts (ECMWF).

- In the past we integrated XIOS into OpenIFS to address the former inefficient sequential I/O scheme (Yepes-Arbós et al.,2022).

OpenIFS-XIOS integration scheme



What factors affect XIOS performance?

There are several **factors** that can be tuned to **directly improve** the XIOS performance:

- Number of servers
- Number of dedicated nodes for servers
- ‘one_file’ vs. ‘multiple_file’ mode
- Size of communication buffers
- 2-level server mode in combination with ‘timeseries’
- Lustre striping (if the Lustre filesystem is used)

XIOS resources

- These two factors are critical to make XIOS scalable:
  - Number of XIOS servers.
  - Number of dedicated nodes for XIOS servers.
- Having more XIOS nodes increases the **bandwidth** between model processes and servers, which is necessary to perform an asynchronous and fast transfer.
- Having more XIOS servers increases the **computational power** on server side (beneficial depending on the post-processing operation such as NetCDF compression), but:
  - Makes the ‘one_file’ mode slower.
  - Data is spread across more NetCDF files if ‘multiple_file’ mode is used (see 2-level server mode).

```
xios:
    xml_dir: "${general.esm_namelist_dir}/oifs/43r3/xios/"
    with_model: oifs
    nproc: 1
    omp_num_threads: 48
```

‘one_file’ vs. ‘multiple_file’ mode

- **'one_file'** mode has a limited computational efficiency as it **does not scale well** when outputting a large volume of data for **high resolution** configurations.
- **'multiple_file'** mode achieves a good computational efficiency as it **scales** with many resources. However, each XIOS server writes its own NetCDF file, so **output data is splitted** between all these partial files (see 2-level server mode).

```
<file_group
    type="multiple_file"
    format="netcdf4"
    par_access="collective"
    name="awi3_atm"
    split_freq="1y">
```

Buffer size settings

There are two parameters to control the buffer size to send data between clients and servers

- '**optimal_buffer_size**': it controls whether using asynchronous or synchronous communications:
  - '**performance**': it uses as much memory as it is needed to bufferize all data between two output periods, so it is the fastest option.
  - '**memory**': it uses the minimum amount of memory needed, so no performance at all.
- '**buffer_size_factor**': XIOS automatically computes the size of the buffers. However, users can adjust it using a multiplying factor.

```
<variable_group id="buffer" > <!-- Tune both "buffer" variables for performance purposes -->
  <variable id="optimal_buffer_size" type="string"> performance </variable>
  <variable id="buffer_size_factor"  type="double"> 1.0         </variable>
</variable_group>
```

2-level server mode

- Level 1: They are in charge of **receiving** the data from OpenIFS processes and redistributing it to subsets of level-two servers (called pools).
- Level 2: They are in charge of **writing** NetCDF files that contain the entire domain into the storage system.
- When enabling 'timeseries' with 2-level server mode and one second level server per pool:
  - Each field is written into a different NetCDF file which contains the entire domain of a field.
  - Files are well-balanced across all second level servers.

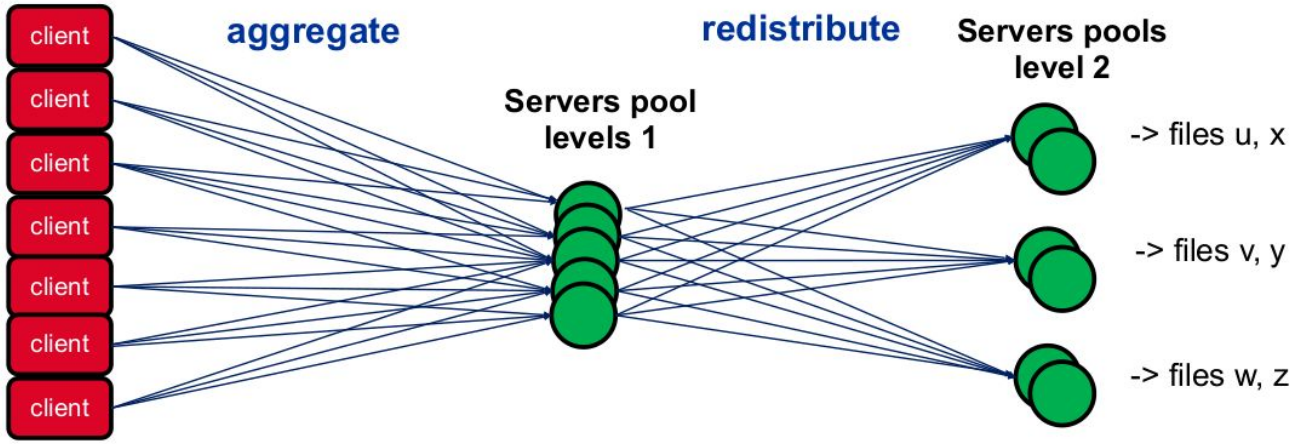


Figure source: XIOS team

Lustre filesystem

- The **Lustre** filesystem stores a file in one or more Object Storage Target (OST) devices.

- If OpenIFS is run on a cluster that uses Lustre it is important to pay attention to the **striping**, which allows to divide a file into chunks that are stored in different OSTs.
  - When using the 'one_file' mode, it is important to set up a striping for each NetCDF at least as equal as to the number of XIOS servers.
  - This allows each XIOS server to write into a different OST, which prevents to affect the performance of the whole system.

Other performance considerations

There are other **factors** that can **implicitly impact** on the XIOS performance:

- Compiler optimization flags
- MPI placing
- Output size
- Output frequency
- Arithmetic and temporal operations such as averages
- Spatial operations such as remapping

XIOS performance reports

- XIOS can generate **performance reports** for each client and server at the end of the execution.
- The client ones are really important to know if OpenIFS processes are **blocked waiting** for the send buffer to be freed.
- The waiting ratio should be **close to zero**.

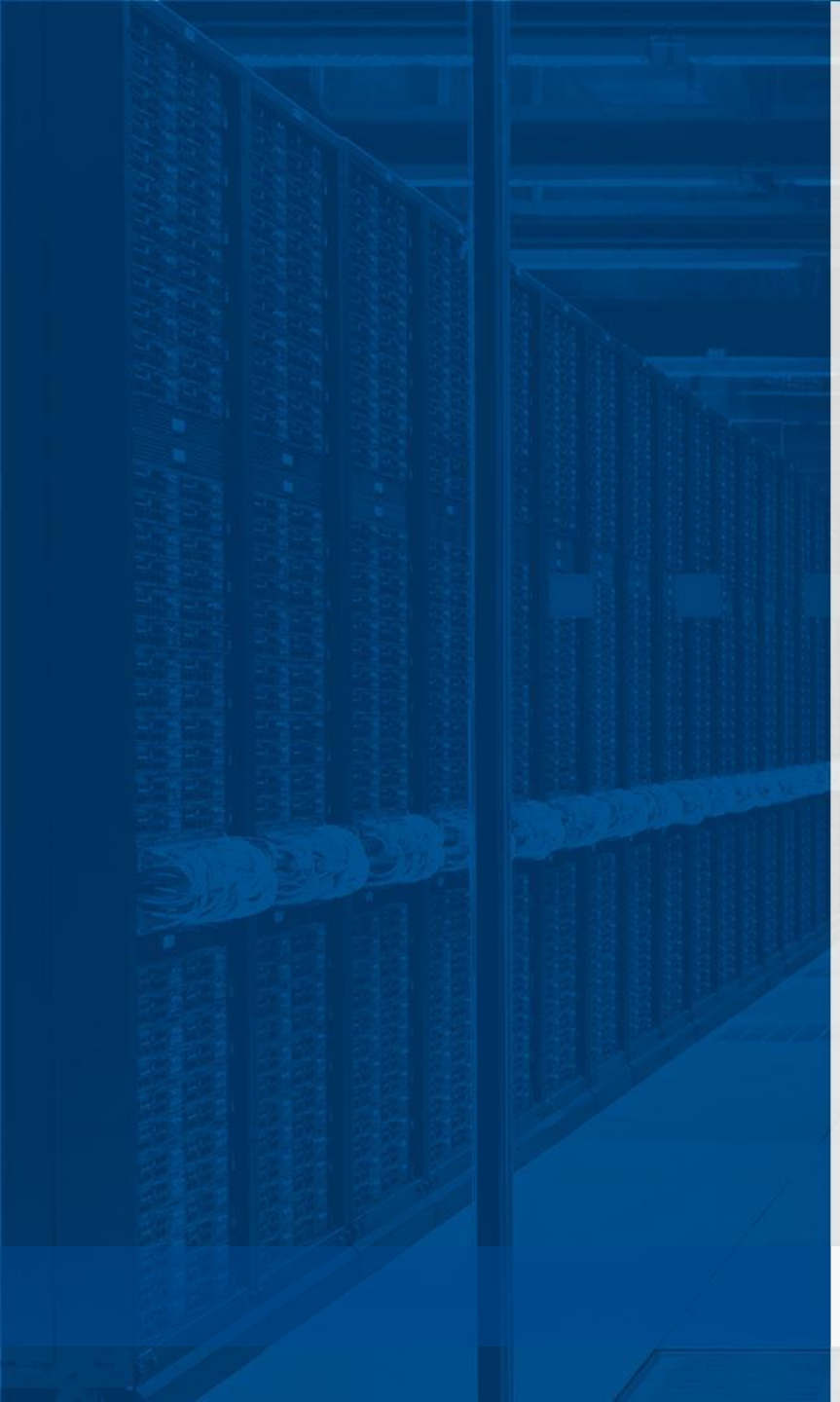
```
-> report :  Performance report : Whole time from XIOS init and finalize: 150.221 s
-> report :  Performance report : total time spent for XIOS : 25.3604 s
-> report :  Performance report : time spent for waiting free buffer : 0.344329 s
-> report :  Performance report : Ratio : 0.229215 %
-> report :  Performance report : This ratio must be close to zero. Otherwise it may be usefull to increase buffer size or numbers of server
-> report :  Memory report : Minimum buffer size required : 80476 bytes
-> report :  Memory report : increasing it by a factor will increase performance, depending of the volume of data wrote in file at each time step of the file
```

- The server ones are also important. The ratio should not be more than 60%.

```
-> report : Performance report : Time spent for XIOS : 143.277
-> report : Performance report : Time spent in processing events : 10.2116
-> report : Performance report : Ratio : 7.12718%
```

Conclusions

- It is possible and necessary to find a **proper setup** for XIOS to achieve a good throughput using an adequate consumption of computational resources. Tuning example on Juwels:
  - Tco95L91 (100 km) - CORE2: From 92 to 134 SYPD
  - Tco159L91 (61 km) - CORE2: From 14 to 64 SYPD
- XIOS (with OpenIFS) has been **tuned** and efficiently **deployed** on the following platforms using different model configurations:
  - MN4: Lenovo based on Intel and GPFS
  - ECMWF's HPCF: Cray XC40 based on Intel and Lustre
  - JUWELS: Atos based on Intel and GPFS
  - Aleph: Cray XC50 based on Intel and Lustre
  - HLRN-IV: Atos based on Intel and DDN Lustre
- Although the OpenIFS-XIOS integration was developed on a specific platform, these results suggest that it is **portable** to different ones.



Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Thank you

esiwace
CENTRE OF EXCELLENCE IN SIMULATION OF WEATHER AND CLIMATE IN EUROPE

xavier.yepes@bsc.es