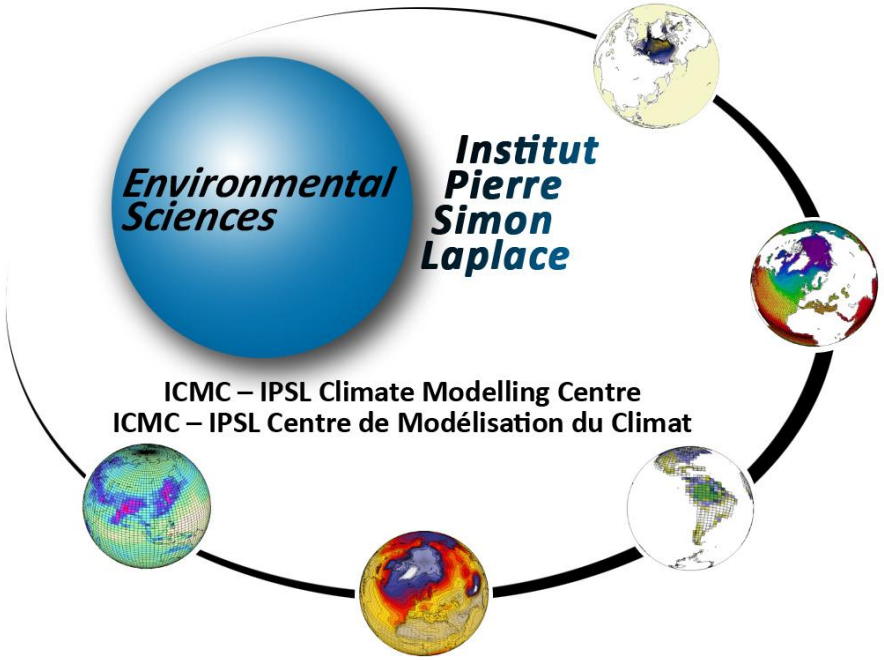
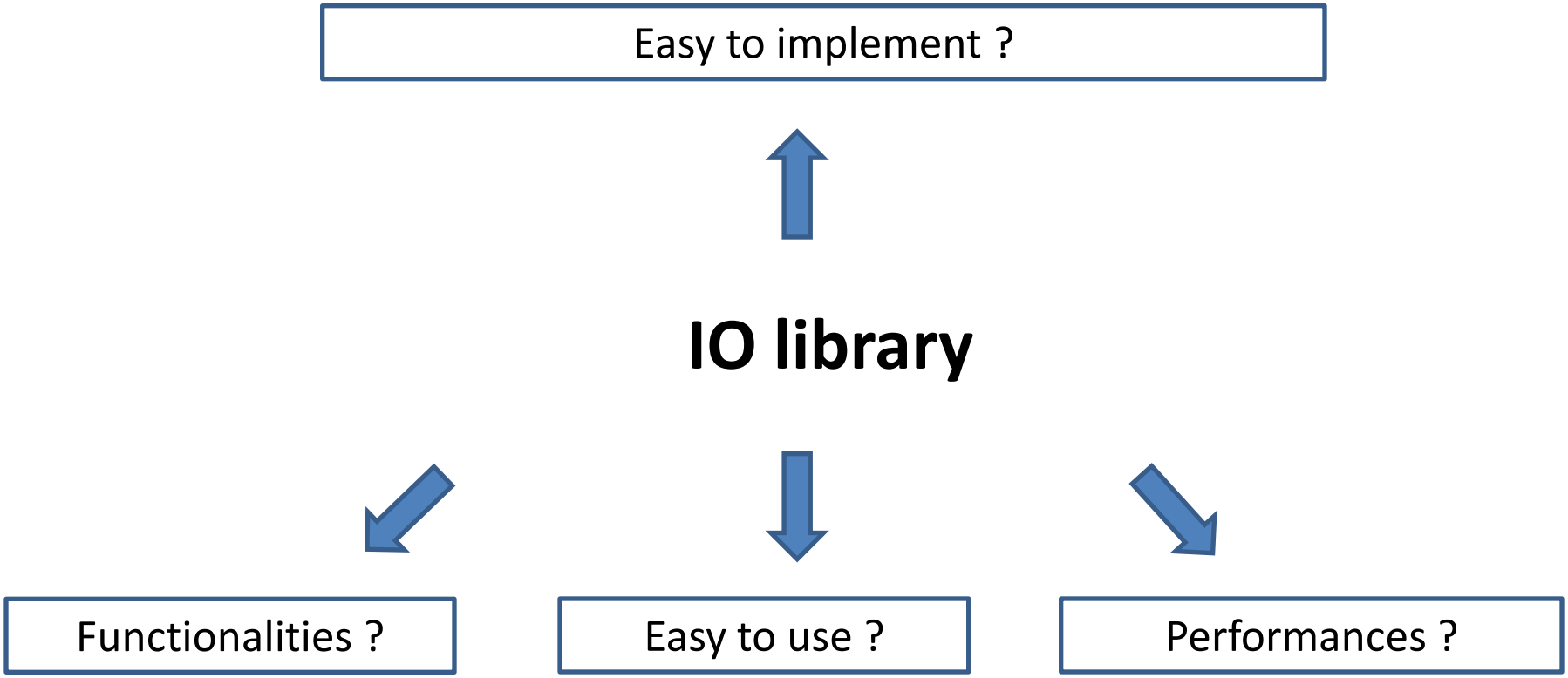


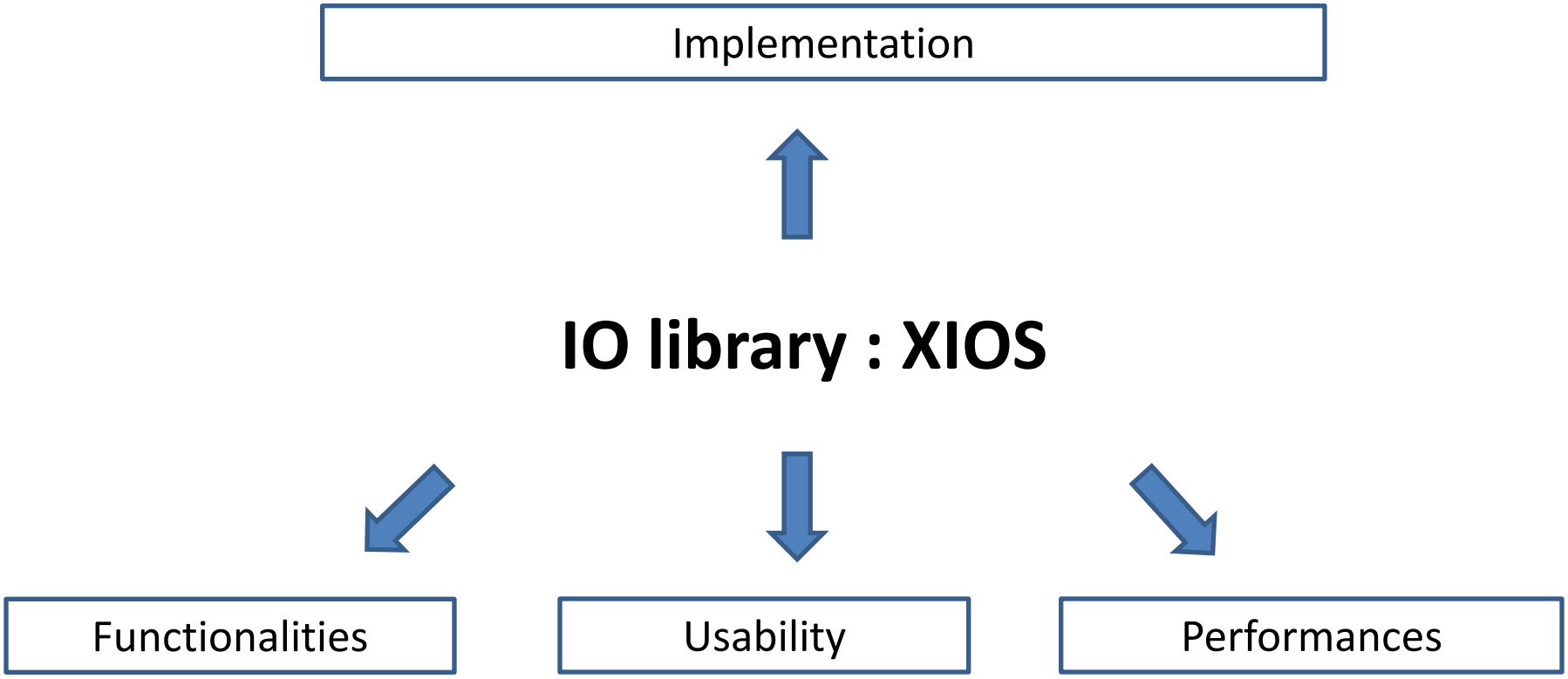
I/O : issues related to implementation (XIOS)

Arnaud Caubel (IPSL)
S. Masson (LOCEAN-IPSL), E. Maisonnave (Cerfacs)

June 12th 2014, IS-ENES2 GA

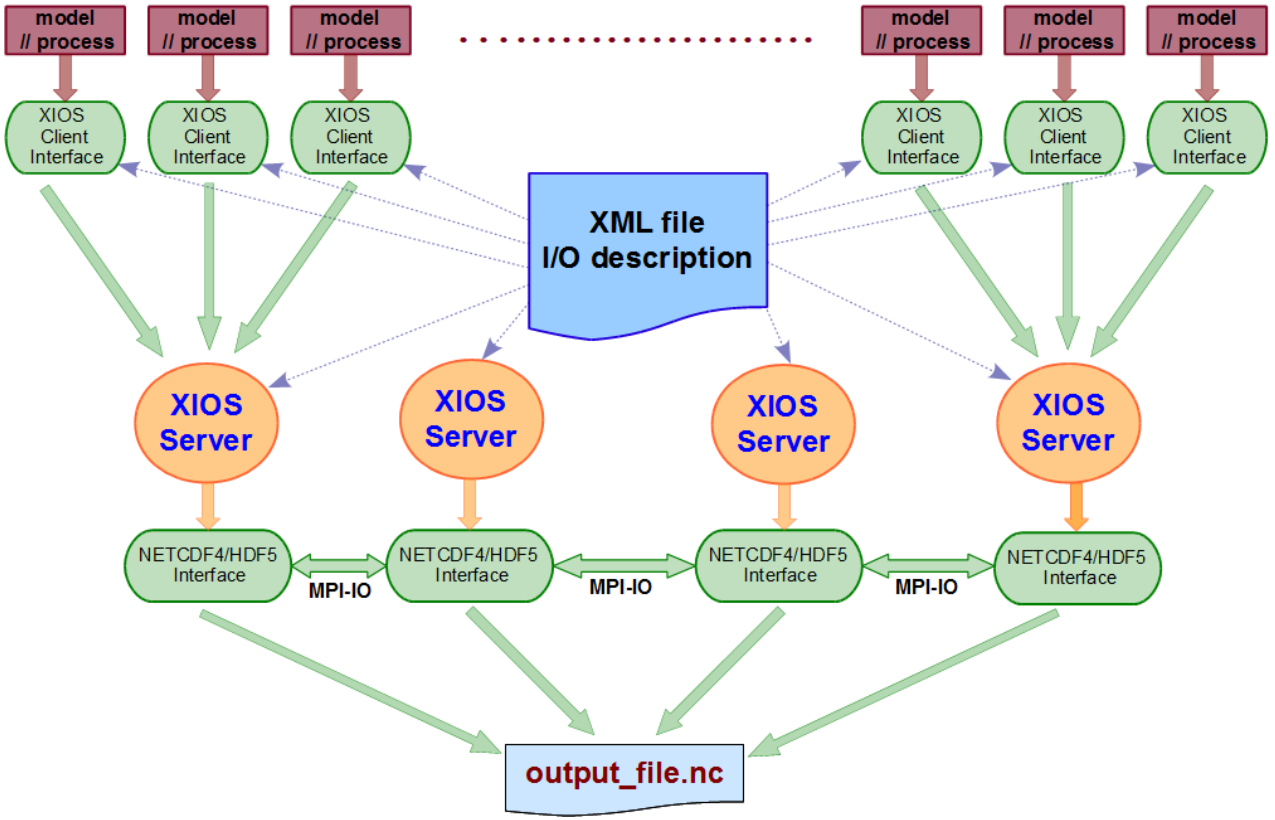

Easy to implement ?
IO library
Functionalities ?
Easy to use ?
Performances ?

XIOS : implementation and user feedback



XIOS in some words

- library dedicated to **IO management of climate code**, developped at **IPSL** by Y. Meurdesoif
- **XML** configuration file
- **attached** mode (library) or **server** mode (asynchroneous transfer), **multiple** (sequential writing) or **single** (parallel writing) output file
- **NetCDF** format (GRIB2 in progress, ICHEC collaboration)



XIOS : implementation

Done

- IPSL components
  - **NEMO,** Oceanic model, curvilinear grid, S. Masson (LOCEAN-IPSL)
  - **LMDZ ,** Atmospheric model, regular grid, student for 3 months training period
  - **ORCHIDEE**, Land scheme model, indexed grid, A. Caubel and J. Ghattas (IPSL)
  - **DYNAMICO,** Atmospheric dynamical core, icosahedral grid, Y. Meurdesoif (IPSL)

- Other components
  - **MAR**, Atmospheric Regional model MAR, H. Gallee (LGGE)
  - **ROMS,** Oceanic regional model ROMS, A. Ponte (IFREMER)
  - **MARS3D**, Model for Applications at Regional Scale, S. Theetten (IFREMER)

- Earth System model
  - **IPSLCM6**, IPSL Earth system model (all components with XIOS), A. Caubel (IPSL)
  - Other Earth system models with NEMO component (EC-Earth)

In progress

- **EC-Earth :** IFS, Atmospheric model, gaussian grid, ICHEC, SMHI
- **CNRM** Earth System model : Arpege/Surfex/Gelato-NEMO (ANR CONVERGENCE project)

XIOS : usability

- Possible implementation by **non-expert developer** despite of “very light existing documentation” : researcher, technical people, student,…
- **Simplification** of the IO management into the code

```
CALL xios_initialize("nemo", return_comm=comm)
CALL xios_context_initialize("nemo_context",comm)
! Grid  definition
…
CALL xios_send_field("sst", sst)
…
CALL xios_context_finalize()
CALL xios_finalize()
```

nemo source code

- **Outsourcing** the output definition in **XML configuration file** with possible splitting.

```
…
<context id="nemo_context"  src="nemo.xml">
…
```

iodef.xml configuration file

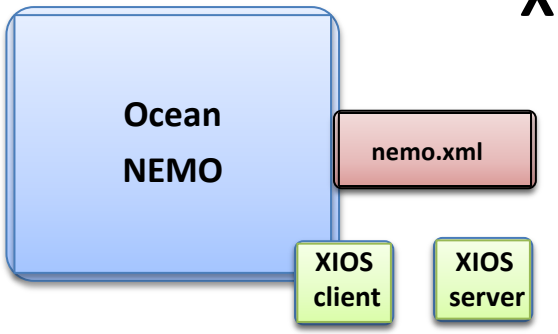
```
<context id="nemo_context">
<field definition src="field_definition_nemo.xml">
…
<\context>
```

nemo.xml configuration file

```
<field definition>
<field id="sst" operation="average" />
…
</field definition>
```

field_definition.xml configuration file

XIOS : Implementation in component



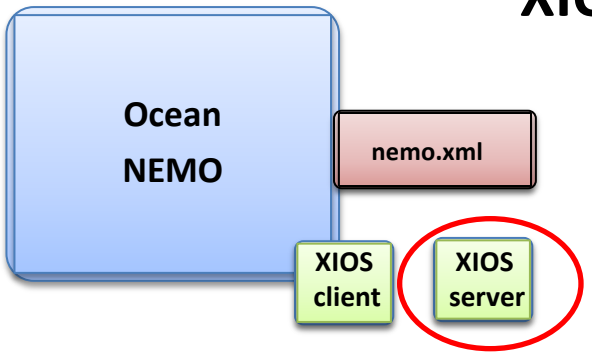

```
CALL xios_initialize("nemo", return_comm=comm)
CALL xios_context_initialize("nemo_context",comm)
…
CALL xios_context_finalize()
CALL xios_finalize()
```

nemo source code

```
<context id="nemo_context"  src="nemo.xml">


<context id="xios">
<variable id="using_server" type="boolean">true</variable>
<variable id="using_oasis" type="boolean">false</variable>
<variable id="oasis_codes_id" type="string">nemo</variable>
<\context>
```

iodef.xml
configuration file

XIOS : Implementation in component




```
CALL xios_initialize("nemo", return_comm)
CALL xios_context_initialize("nemo_context", comm)
...
CALL xios_context_finalize()
CALL xios_finalize()
```

nemo source code

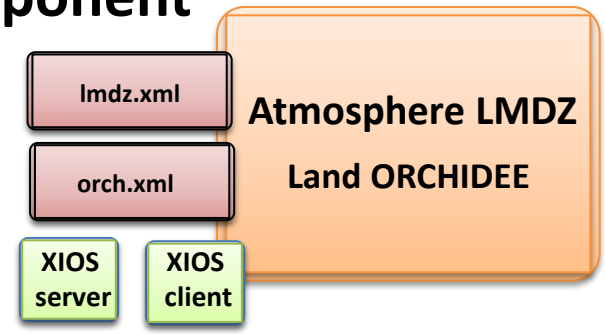
Forced configuration
Server mode

```
<context id="nemo_context"  src="nemo.xml">


<context id="xios">
<variable id="using_server" type="boolean">true</variable>
<variable id="using_oasis" type="boolean">false</variable>
<variable id="oasis_codes_id" type="string">nemo</variable>
<\context>
```

iodef.xml
configuration file

link as a library : with ***-lstdc++ -lxios***
launch command : ***mpirun –np 10 nemo.exe : -np 1 xios.exe***

XIOS : Implementation in component




```
CALL xios_initialize("lmdz", return_comm=comm)
CALL xios_context_initialize("lmdz_context",comm)
CALL xios_context_initialize("orch_context",comm)
…
CALL xios_context_finalize()
CALL xios_context_finalize()
CALL xios_finalize()
```

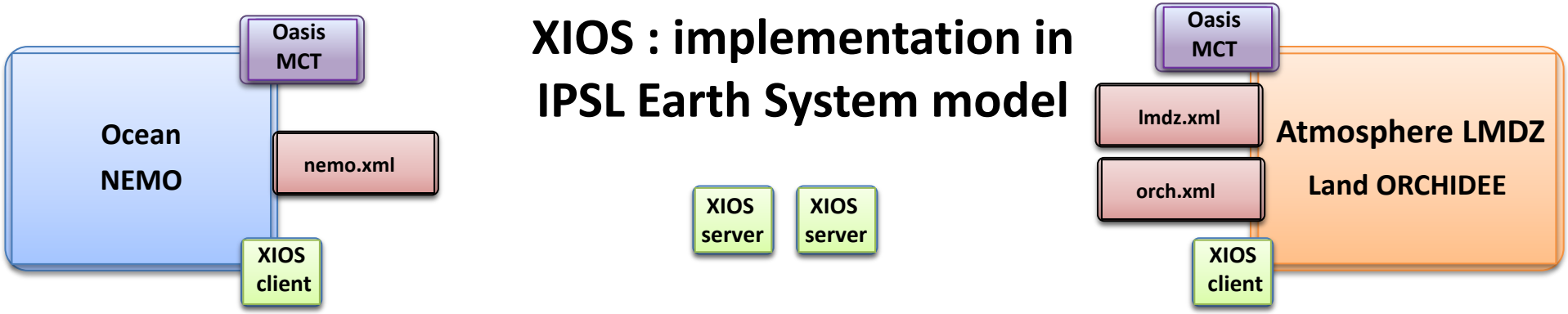
lmdz/orchidee source code

```
<context id="lmdz_context"  src="lmdz.xml">

<context id="orch_context"  src="orch.xml">


<context id="xios">
<variable id="using_server" type="boolean">true</variable>
<variable id="using_oasis" type="boolean">false</variable>
<variable id="oasis_codes_id" type="string">lmdz</variable>
<\context>
```

iodef.xml
configuration file

Forced configuration
mpirun –np 10 lmdz.exe : -np 1 xios.exe

XIOS : implementation in IPSL Earth System model




```
CALL xios_initialize("nemo", return_comm=comm)
CALL xios_context_initialize("nemo_context",comm)
…
CALL xios_context_finalize()
CALL xios_finalize()
```

nemo source code

```
CALL xios_initialize("lmdz", return_comm=comm)
CALL xios_context_initialize("lmdz_context",comm)
CALL xios_context_initialize("orch_context",comm)
…
CALL xios_context_finalize()
CALL xios_context_finalize()
CALL xios_finalize()
```

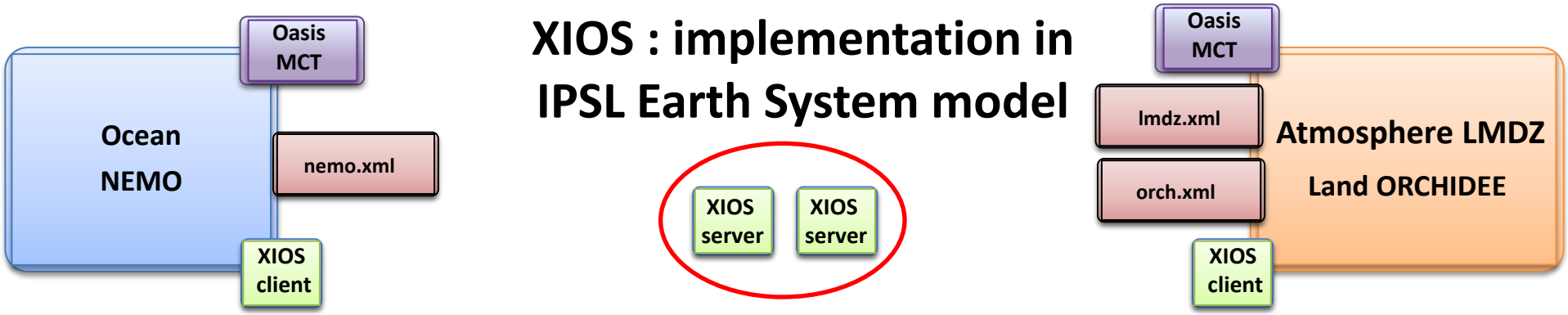
lmdz/orchidee source code

```
<context id="nemo_context"  src="nemo.xml">

<context id="lmdz_context"  src="lmdz.xml">

<context id="orch_context"  src="orch.xml">


<context id="xios">
<variable id="using_server" type="boolean">true</variable>
<variable id="using_oasis" type="boolean">true</variable>
<variable id="oasis_codes_id" type="string">nemo,lmdz</variable>
<\context>
```

iodef.xml
configuration file

mpirun –np 10 nemo.exe : -np 10 lmdz.exe : -np 2 xios.exe

XIOS : implementation in IPSL Earth System model




```
CALL xios_initialize("nemo", return_comm=comm)
CALL xios_context_initialize("nemo_context",comm)
…
CALL xios_context_finalize()
CALL xios_finalize()
```

nemo source code

```
CALL xios_initialize("lmdz", return_comm=comm)
CALL xios_context_initialize("lmdz_context",comm)
CALL xios_context_initialize("orch_context",comm)
…
CALL xios_context_finalize()
CALL xios_context_finalize()
CALL xios_finalize()
```

lmdz/orchidee source code

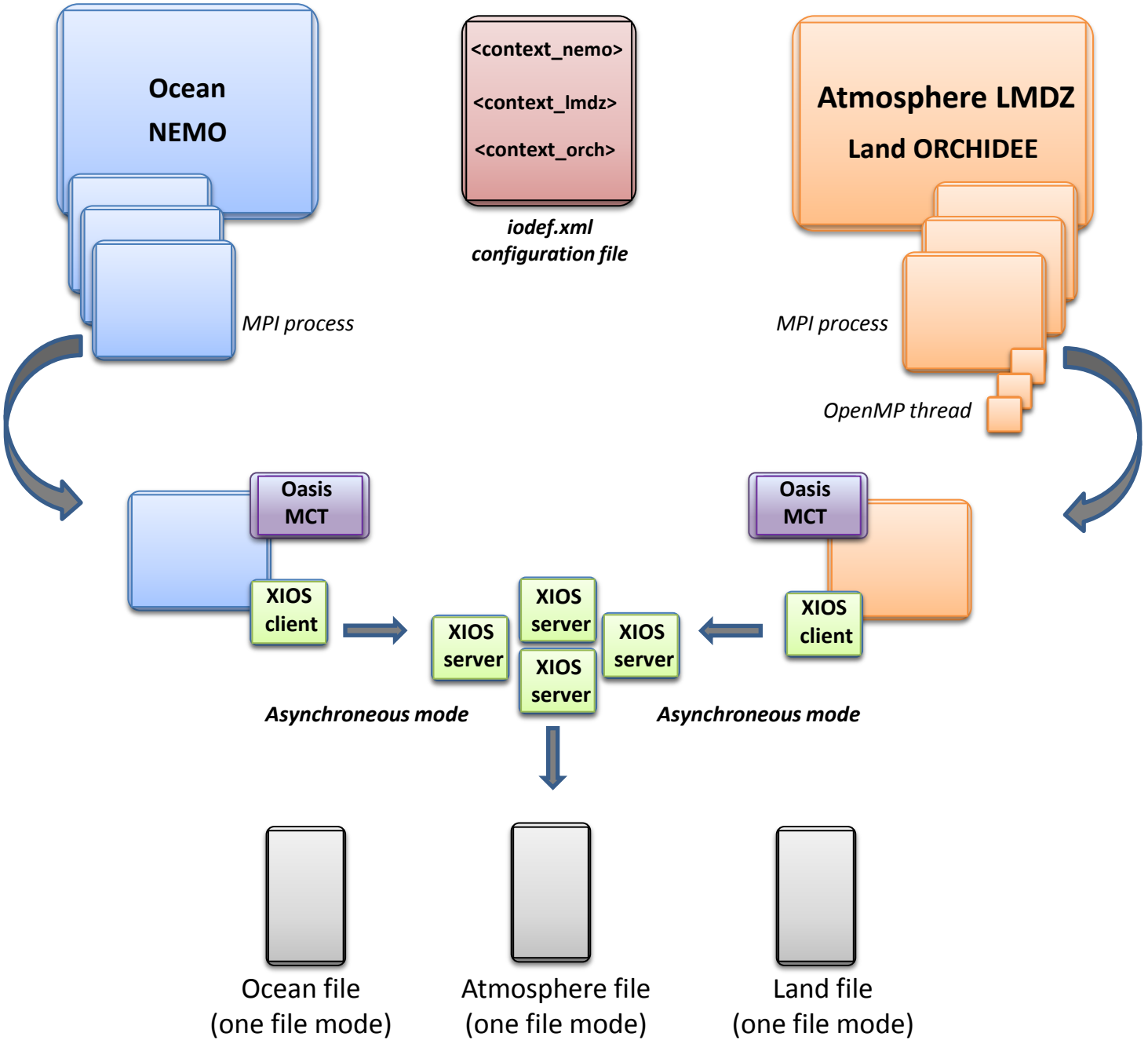
Coupled configuration
Server mode

```
<context id="nemo_context"   src="nemo.xml">

<context id="lmdz_context"   src="lmdz.xml">

<context id="orch_context"   src="orch.xml">


<context id="xios" >
<variable id="using_server" type="boolean">true</variable>
<variable id="using_oasis" type="boolean">true</variable>
<variable id="oasis_codes_id" type="string">nemo,lmdz</variable>
<\context>
```

iodef.xml
configuration file

mpirun –np 10 nemo.exe : -np 10 lmdz.exe : -np 2 xios.exe

XIOS : implementation in IPSL Earth System model



XIOS : functionalities

- **Output format** : NetCDF, GRIB2 (in progress, collaboration ICHEC)
- **Use of parallel writing :**
  - Allows to obtain one output single file
  - Step of output files recombination not needed anymore : running environment more reliable.
- **Operations** on variables
  - arithmetic field combination

```
<field id="sst" long_name="Sea Surface Temperature " unit="degC" prec="8"/>

<field name="tosK" unit="degK" > sst+273.15 </field >
<field name="tostos" > sst * sst</field >
```

  - change of precision (ex : integer 2 with add_offset and scale_factor attributes)

```
<field field_ref="sss" name="sos_i2" prec="2" add_offset="20." scale_factor="1.e-3" />
```

  - personal NetCDF file/field attributes

```
<field field_ref="sst"      name="tos" >
  <variable id="my_attribute1" type="string" >blabla</variable>
```

XIOS : performances

- **Server** mode
  - **asynchroneous transfer** between XIOS client (model) and XIOS server
  - Allows to reduce (remove ?) **impact of the I/O** on the writing of data
  - One server allows to have one output file
    - IPSL Earth system model : first tests on Curie to write ocean and atmospheric output data : **no significative loss of performance** in comparaison with no IOs.
    - depends on the amount of data to write (**output level**)
  - Several servers to avoid to fill XIOS buffers
    - Arpege-NEMO coupled model on MareNostrum, with **16 XIOS servers** for 1024 computing NEMO process (**no IO impact**).

- **Parallel writing**
  - several processes (XIOS clients or servers) write into **one single output file**
  - Machine dependent
    - **“filesystem dependent”**
      - Need to test parameters of parallel filesystem (striping for Lustre,…)
      - Good performances obtained with appropriate parameters
    - implementation of **parallel** library NetCDF4/HDF5 on the machine ?

Conclusions

- **easy to implement XIOS** but better to implement step by step :
    - Computing parallelism : sequential -> parallel (MPI) -> hybrid parallelism (MPI + OpenMP)
    - Parallel writing : multiple file -> single file (parallel writing)
    - Transfer mode : attached mode (XIOS as a library) -> server mode
    - Forced configuration -> coupled configuration (using OASIS-MCT coupler)

- easy to switch from forced to **coupled configuration**

- **possibility** to **reach good performances** …but very « machine-dependant »

- **XIOS more and more used**, not only at IPSL

- **XIOS users are** interested in :
    - more **documentation** (related to XIOS library functionalities)
    - more **functionalities** :
        - on line post-treatment : grid remapping, interpolations, zonal means , grid coarsening, downscaling, global/partial reduction, time series production at runtime
        - parallel reading (forcing files,…)
    - handling easily (or more easily than CMIP5 !) **CMIP6 outputs** requirements