XIOS in ORCHIDEE
and how to control output level

Thanks to
Arnaud Cauel, first implementation of XIOS in ORCHIDEE
Yann Meurdesoif, main developer of XIOS

Short presentation by Josefine Ghattas
ORCHIDEE Training course 24 november 2016

Implementation in ORCHIDEE

- Implementation in ORCHIDEE done by Arnaud Caubel and Josefine Ghattas, introduced in ORCHIDEE **trunk revision 1788**, June 2014
- Results validated against IOIPSL output at curie
- Running at **curie**/TGCC, **ada**/IDRIS, **ciclad/climserv**/IPSL and **obelix**/LSCE
- To be used with libIGCM configurations:

  ORCHIDEE_trunk, **LMDZOR_v6** and **IPSLCM6**

- In ORCHIDEE:
  - src_parallel/**xios_orchidee.f90** : module doing all interfacing to XIOS
  - **src_xml** : directory containing xml files for running with XIOS
  - Parameter **XIOS_ORCHIDEE_OK** in to activate running with XIOS
  - Preprocessing key XIOS to enabel linking to XIOS

xios_orchidee_send_field

Example from thermosoil_main:

```
USE xios_orchidee

REAL(r_std),DIMENSION (kjpindex)         :: soilflx
REAL(r_std),DIMENSION (kjpindex)         :: surfheat_incr
REAL(r_std),DIMENSION (kjpindex, ngrnd) :: ptn
...

CALL xios_orchidee_send_field("ptn",ptn)
CALL xios_orchidee_send_field("Qg",soilflx)
CALL xios_orchidee_send_field("DelSurfHeat",surfheat_incr)
```

Syntax: **CALL xios_orchidee_send_field(field_id, field)**

field_id: a unique identifier, the same id is set in the field definition in parameter file field_def_orchidee.xml which must be present at run time
CHARACTER(len=*)

field: the variable to send to XIOS. The variable is on landpoint grid, it can have 1 or 2 supplementary axis:
REAL(r_std), DIMENSION(kjpindex) or
REAL(r_std), DIMENSION(kjpindex,:)

Convention in ORCHIDEE : use the same name for the id as the variable name

xml parameter files

To run ORCHIDEE with XIOS all diagnostic output files are configured thourgh xml files. Follwing 4 files needs to be present at each execution :

- **iodef.xml** Main input file for XIOS
- **context_orchidee.xml** Axis and domain information, include field and file def
- **field_def_orchidee.xml** **=> Definition for each variable send in ORCHIDEE**
  => Only change if added new variable in ORCHIDEE
- **file_def_orchidee.xml** **=> Specify all output files and there variables**
  => Change to set your output level
  => Remove variables, change levels, change freq...

The above xml file are stored in ORCHIDEE/src_xml directory.

file_def_orchidee.xml

```
<!-- ============================================================================================== -->
<!-- file_def_orchidee.xml : Definition of output files                                             -->

<file_definition type="one_file" par_access="collective" enabled=".TRUE." min_digits="4">

  <!-- Sechiba file 1 -->
  <file id="sechiba1" name="sechiba_history" output_level="11" output_freq="1d" enabled=".TRUE.">
    <field field_ref="Areas" level="1"/>
    <field field_ref="LandPoints" level="1"/>
    <field field_ref="Contfrac" level="1"/>
    <field field_ref="evap" level="1"/>
    <field field_ref="coastalflow" level="1"/>
    <field field_ref="riverflow" level="2"/>
    <field field_ref="temp_sol_C" level="2"/>
    ...
  </file>

  <!-- Sechiba file 2 -->
  <file id="sechiba2" name="sechiba_out_2" output_level="2" output_freq="1d" enabled=".TRUE.">
    <field field_ref="Areas" level="1"/>
    <field field_ref="LandPoints" level="1"/>
    <field field_ref="Contfrac" level="1"/>
    <field field_ref="mrsos" level="1"/>
    <field field_ref="mrro" level="2"/>
    ...
  </file>

  <!-- Stomate file 1 -->
  <file id="stomate1" name="stomate_history" output_level="10" output_freq="86400s">
    <field field_ref="RESOLUTION_X" level="1"/>
    <field field_ref="RESOLUTION_Y" level="1"/>
    <field field_ref="CONTFRAC_STOMATE" level="1"/>
  </file>
</file_definition>
```

file_def_orchidee.xml

```
<!-- ============================================================================================== -->
<!-- file_def_orchidee.xml : Definition of output files                                             -->

<file_definition type="one_file" par_access="collective" enabled=".TRUE." min_digits="4">

  <!-- Sechiba file 1 -->
  <file id="sechiba1" name="sechiba_history" output_level="11" output_freq="1d" enabled=".TRUE.">
    <field field_ref="Areas" level="1"/>
    <field field_ref="LandPoints" level="1"/>
    <field field_r
    <field field_r
    <field field_r
    <field field_r
    <field field_r
    ...
  </file>

  <!-- Sechiba file 2 -->
  <file id="sechiba2" name="sechiba_out_2" output_level="2" output_freq="1d" enabled=".TRUE.">
    <field field_ref="Areas" level="1"/>
    <field field_ref="LandPoints" level="1"/>
    <field field_ref="Contfrac" level="1"/>
    <field field_ref="mrsos" level="1"/>
    <field field_ref="mrro" level="2"/>
    ...
  </file>

  <!-- Stomate file 1 -->
  <file id="stomate1" name="stomate_history" output_level="10" output_freq="86400s">
    <field field_ref="RESOLUTION_X" level="1"/>
    <field field_ref="RESOLUTION_Y" level="1"/>
    <field field_ref="CONTFRAC_STOMATE" level="1"/>
  </file>
</file_definition>
```

Information about all files written by ORCHIDEE

type “one_file” or “multiple_file” : XIOS will gather information from all processes on a single output file or not

file_def_orchidee.xml

```
<!-- ================================================================================================== -->
<!-- file_def_orchidee.xml : Definition of output files                                                 -->

<file_definition type="one_file" par_access="collective" enabled=".TRUE." min_digits="4">

  <!-- Sechiba file 1 -->
  <file id="sechiba1" name="sechiba_history" output_level="11" output_freq="1d" enabled=".TRUE.">
    <field field_ref="Areas" level="1"/>
    <field field_ref="LandPoints" level="1"/>
    <field field_ref="Contfrac" level="1"/>
    <field field_ref="evap" level="1"/>
    <field field_ref="coastalflow" level="1"/>
    <field field_ref="riverflow" level="2"/>
    <field field_ref="temp_sol_C" level="2"/>
    ...
  </file>

  <!-- Sechiba file 2 -->
  <file id="sechiba2" name="sechiba_out_2" output_level="2" output_freq="1d" enabled=".TRUE.">
    <field field_ref="Areas" level="1"/>
    <field field_ref="LandPoints" level="1"/>
    <field field_ref="Contfrac" level="1"/>
    <field field_ref="mrsos" level="1"/>
    <field field_ref="mrro" level="2"/>
    ...
  </file>

  <!-- Stomate file 1 -->
  <file id="stomate1" name="stomate_history" output_level="10" output_freq="86400s">
    <field field_ref="RESOLUTION_X" level="1"/>
    <field field_ref="RESOLUTION_Y" level="1"/>
    <field field_ref="CONTFRAC_STOMATE" level="1"/>
  </file>
</file_definition>
```

file_def_orchidee.xml

```
<!-- ============================================================================================== -->
<!-- file_def_orchidee.xml : Definition of output files                                             -->

<file_definition type="one_file" par_access="collective" enabled=".TRUE." min_digits="4">

  <!-- Sechiba file 1 -->
  <file id="sechiba1" name="sechiba_history" output_level="11" output_freq="1d" enabled=".TRUE.">
    <field field_ref="Areas" level="1"/>
    <field field_ref="LandPoints" level="1"/>
    <field field_ref="Contfrac" level="1"/>
    <field field_ref="evap" level="1"/>
    <field field_ref="coastalflow" level="1"/>
    <field field_ref="riverflow" level="2"/>
    <field field_ref="temp_sol_C" level="2"/>
    ...
  </file>

  <!-- Sechi
  <file id="
    <field f
    <field f
    <field f
    <field f
    <field f
    ...
  </file>

  <!-- Stoma
  <file id="
    <field f
    <field f
    <field f
  </file>
</file_defin
```

Information line about one file

name	filename, suffix .nc will be added to the filename
output_level	**“x”** : all variables listed below with level less or equal to x will be written
output_freq	”1d”, “1800s”, “1ts”, “1mo”, “3h”, “1y” : frequency for the file
enabled	“.TRUE.” / “.FALSE.” : create the file, true is default

file_def_orchidee.xml

```
<!-- ============================================================================================== -->
<!-- file_def_orchidee.xml : Definition of output files                                             -->

<file_definition type="one_file" par_access="collective" enabled=".TRUE." min_digits="4">

  <!-- Sechiba file 1 -->
  <file id="sechiba1" name="sechiba_history" output_level="11" output_freq="1d" enabled=".TRUE.">
    <field field_ref="Areas" level="1"/>
    <field field_ref="LandPoints" level="1"/>
    <field field_ref="Contfrac" level="1"/>
    <field field_ref="evap" level="1"/>
    <field field_ref="coastalflow" level="1"/>
    <field field_ref="riverflow" level="2"/>
    <field field_ref="temp_sol_C" level="2"/>
    ...
  </file>

  <!-- Sechiba
  <file id="se
    <field fie
    <field fie
    <field fie
    <field fie
    <field fie
    ...
  </file>

  <!-- Stomate
  <file id="st
    <field fie
    <field fie
    <field fie
  </file>
</file_definit
```

A line per variable added in the file

field_ref	reference id as set in field_def_orchidee.xml file
level	**“x”** : the variable is only written if this level is less or equal of output_level set at the file description line above.
name / long_name	”new_name” : name of the variable in the output file. If it is not set, the name set in field_def_orchidee.xml will be used.
enabled	“.TRUE.” / “.FALSE.” : write the variable, true is the default.
operation	can be added, overwrites settings in field_def “average”, “min”, “max”, “instant”

Add a new variable in ORCHIDEE

1) Add in the ORCHIDEE module where the variable is calculated:

```
CALL xios_orchidee_send_field("new_var",new_var)
```

2) In field_def_orchidee.xml, add declaration of the variable

3) In file_def_orchidee.xml : add the variable in all file sections where you want to write it

-) If the variable is only calculated for a specific option, add an exception in xios_orchidee_init. This avoid that the variable will be initialized in the output file without being written if you keep the same .xml files.

Running in attached mode

Requirements for running ORCHIDEE with XIOS in attached mode:

- 1 executable: **orchidee_ol**
- 4 xml files : iodef.xml, context_orchidee.xml,
  field_def_orchidee.xml, file_def_orchidee.xml
- Parameter file: run.def
- Input netcdf files: forcing_file.nc, PFTmap.nc, ...

Change in iodef.xml:

<variable id="**using_server**" type="boolean">**false**</variable>

It it possible to run on 1 or several MPI

Note: You can copy xml files from ORCHIDEE/src_xml

Running with server

Requirements for running ORCHIDEE with XIOS using server:

- 2 executables: **orchidee_ol** and **xios_server.exe**
- 4 xml files : iodef.xml, context_orchidee.xml,
  field_def_orchidee.xml, file_def_orchidee.xml
- Parameter file: run.def
- Input netcdf files: forcing_file.nc, PFTmap.nc, ...

Change in iodef.xml:

<variable id=**"using_server"** type="boolean">**true**</variable>

Note: You can copy xml files from ORCHIDEE/src_xml

Using libIGCM configurations
server mode by default

Default mode is using 1 server XIOS in libIGCM configurations (ORCHIDEE_trunk, LMDZOR_v6, IPSLCM6)

config.card:
- Component IOS reperesents XIOS
- Set number of cores MPI for each executable with 1MPI for the xios server.

```
#============================================================
#D-- ListOfComponents -
[ListOfComponents]
#D- For each component, Name of component, Tag of component
SRF= (sechiba, orchidee_trunk)
SBG= (stomate, orchidee_trunk)
OOL= (orchidee_ol, orchidee_trunk)
IOS= (xios, XIOS)

#D-- Executable -
[Executable]
#D- For each component, Real name of executable
SRF= ("", "")
SBG= ("", "")
OOL= (orchidee_ol, orchidee_ol, 31MPI)
IOS= (xios_server.exe, xios.x, 1MPI)

...

#D-- IOS -
[IOS]
WriteFrequency=""
Restart= n
RestartDate=
RestartJobName=
RestartPath=
```

Number of cores MPI per executable

Using libIGCM configurations
control of output

- field_def_orchidee.xml, file_def_orchidee.xml and context_orchidee.xml are copied from source directory ORCHIDEE/src_xml

- **Use *WriteFrequency* in config.card** to control the frequency and activation of predefined files

  or

- **Change directly in ORCHIDEE/src_xml/file_def_orchidee.xml**

Installing at a new platform

- Currently ORCHIDEE with XIOS has been tested at obelix, ciclad, climserv, curie and ada
- Requirements are MPI and netCDF4 library
- Additional requirements: parallel library NetCDF4/HDF5
  - several processes (XIOS clients or servers) can write into one single output file

Steps to follow for installation at a new platform:

1. Install configuration ORCHIDEE_trunk in a new modipsl

2. Modify compile options in following files:
   - modipsl/util/**AA_make.gdef**
   - modipsl/modeles/**ORCHIDEE/arch/arch-*yourtarget*.[fcm/path]**
   - modipsl/modeles/**XIOS/arch/arch/arch-*yourtarget*.[fcm/path/env]**

   Note: the variable FCM_ARCH in AA_make.gdef is the name of the arch files in ORCHIDEE/arch and XIOS/arch.

3. Recreate makefiles with target chosen above and compile as usual
   cd modipsl/util; ./ins_make -t yourtarget