ICOMEX

_ICO_sahedral-grid _M_odels
for _EX_ascale Earth system simulations

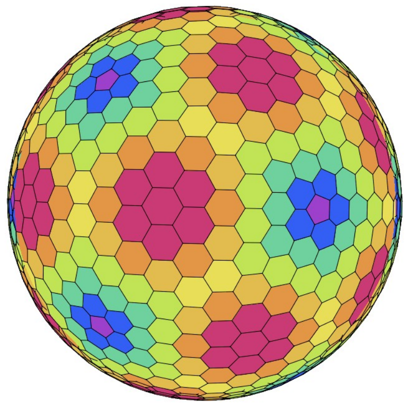
G8 Call
2011-2014, ~1M€ mostly for workforce

Günther Zängl (Deutscher Wetterdienst)
Hirofumi Tomita (RIKEN/AICS)
Masaki Satoh (U. Tokyo)
Thomas Ludwig (U. Hamburg)
Leonidas Linardakis(MPI-M)
John Thuburn(U. Exeter/MetOffice)
Thomas Dubos (IPSL/École Polytechnique)

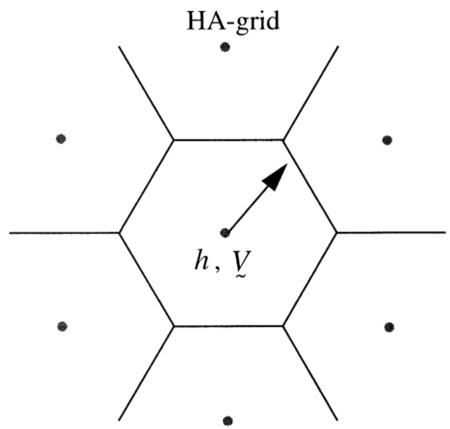
- Members of the consortium and participating models
- General goals and approach of the project
- Specific workpackages
  - *Model intercomparison and evaluation*
  - *Abstract model description scheme*
  - *Feasibility study for using GPUs*
  - *Implicit solvers for massively parallel computing platforms*
  - *Parallel internal postprocessing*
  - *Parallel I/O*
  - *Collaboration with hardware vendors*

Members of the consortium and participating models

All groups have developed / are developing dynamical cores on icosahedral grids. However they differ in terms of numerics and grid structuredness.

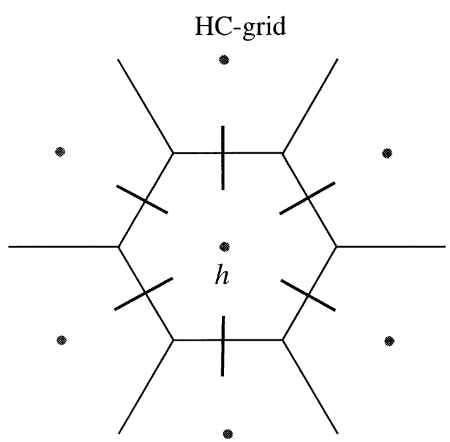


- NICAM : Hirofumi Tomita (RIKEN/AICS), Masaki Satoh(U. Tokyo)
  Development started ~10 years ago
  Special care of moist energy budget
  Structured A-grid



- ICON : Gunther Zaengl (DWD), Marco Giorgetta (MPI-M)
  Development started ~10 years ago
  Unstructured triangular C-grid

- MPAS : John Thuburn (U. Exeter)
  Development started a few years ago
  Special care of energy and vorticity budgets
  Unstructured hexagonal C-grid



- DYNAMICO : T. Dubos (IPSL/École Polytechnique)
  Work really started only a year ago
  Spatial discretization very similar to MPAS, except transport
  Structured hexagonal C-grid

General goals and approach

ICOMEX is not about merging the participating models together...

It is about identifying/adressing roadblocks towards exascale computing :

- Identifying bottlenecks in existing models and platforms
  - *Model intercomparison and evaluation*
  - *Parallel I/O*

- Adressing common, known bottlenecks/challenges, present and future
  - *Abstract model description scheme (domain-specific language)*
  - *Feasibility study for using GPUs*
  - *Implicit solvers for massively parallel computing platforms*
  - *Parallel internal postprocessing*

- Initiating a dialogue with hardware vendors (co-design)
  - *Collaboration with hardware vendors*

Solutions are developed within a specific model but should benefit all partners

Model intercomparison and evaluation
M. Satoh, H. Tomita

Sequence of experiments of increasing length, physical complexity, and relevance to climate modelling

- Deterministic test case
  - *No physics*
  - *Baroclinic wave (Jablonowski & Williamson, 2006)*
  - *Identify/solve implementation (efficiency) issues*

- Statistical test cases
  - *Full physics*
  - *Radiatively-forced general circulation (Held & Suarez, 1994)*
  - *Multi-year aquaplanet experiments (Neale and Hoskins, 2000)*
  - *Check conservation, spectra, wave activity, ...*

- 30-year AMIP run
  - *Experiment 3.3 of the CMIP5 experimental design*
  - *Huge amount of I/O*
  - *'Flagship experiment'*

- Experiments to be performed with all participating models on the K-computer in Tokyo (~10Pflops, 100 000 CPUs)
- Resolution to be defined based on model performance.

Abstract model description scheme for eff cient use of memory bandwidth on a variety of platforms

L. Linardakis

Motivation

- Architectures: wider spectrum, requiring not only different structures but even different languages
- Models: increasing number of software components and code complexity (ICON: Atmo + Ocean, triangle+hexagon, (non-)hydrostatic, nesting, machine specific coding, vectorization, complex orderings,...)
- Not easy to develop/modify/adapt.
- <u>How to create code</u>: a. Compatible and Efficient on different architectures b. Well Engineered

ICON Domain-specific language

L. Linardakis

Proposed Approach

May look like

```
DEFINE OPERATOR div(out_value, in_value, div_coeff)

  REAL, EDGES, 3D, INTENT(in)       :: in_value
  REAL, CELLS, 3D, INTENT(inout)    :: out_value
  REAL, CELLS.EDGES, 2D, INTENT(in):: div_coeff
  INDEX, CELLS, 3D                  :: cell
  INDEX, CELLS.EDGES, 3D            :: edge

  FOR cell IN out_value.cells
    out_value(cell) =
      SUM FOR edge IN cell.edges
        (edge_value(edge) * div_coeff(edge))
      END SUM
  END FOR

END DEFINE div
```

ICON Domain-specific language

L. Linardakis

Proposed Approach

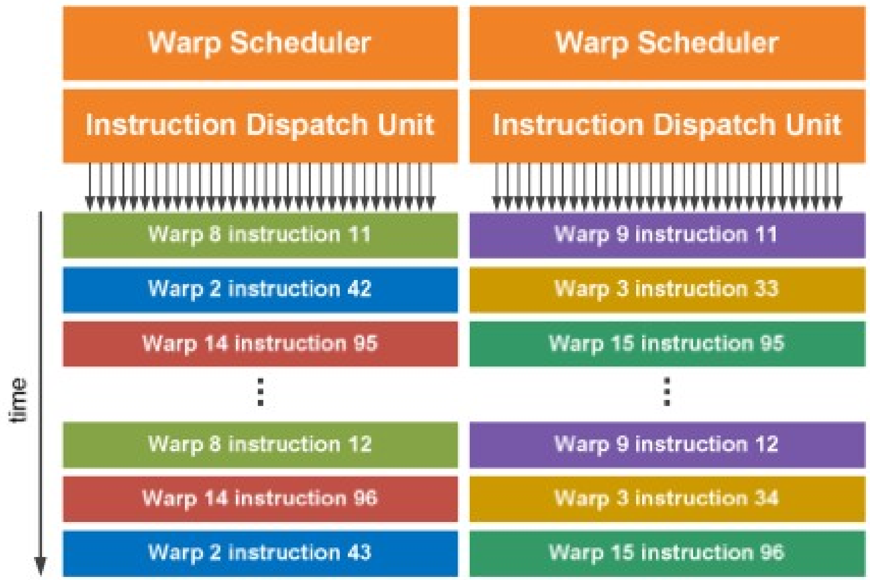
- The actual data structures are hidden – inherited by the grid structure
- Allows translation to different structures by the parser
- Attach static and dynamic metadata to the variables, so that parser and the code know the variable properties
- Object-oriented flavor produces cleaner code

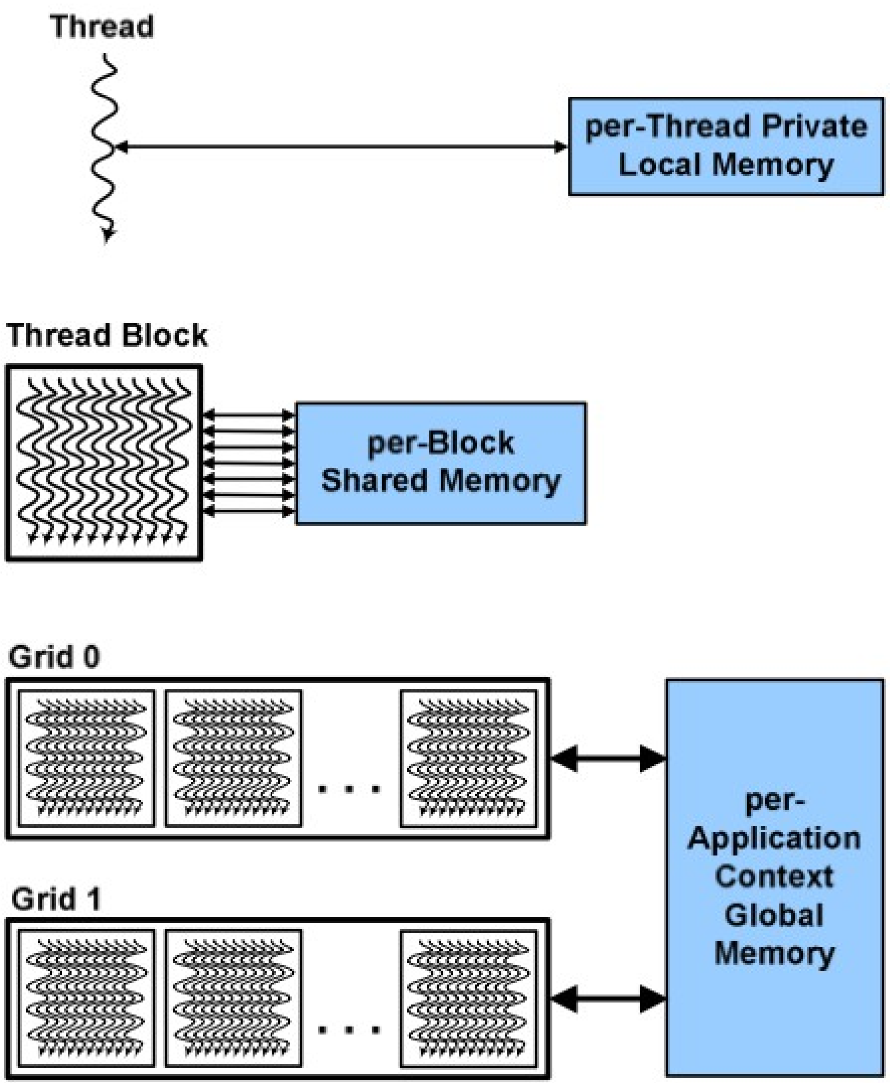
Feasibility for using GPUs for atmospheric models

T. Dubos, Y. Meurdesoif

- *Stack many 'light' cores together*
- *Probable precursor of exascale building block*

- Lock-step execution of 'warps'
- Hates if/else
- Complex memory hierarchy
- Very expensive access to global memory
- Need high compute/data ratio
- Try and reuse data but very little local memory

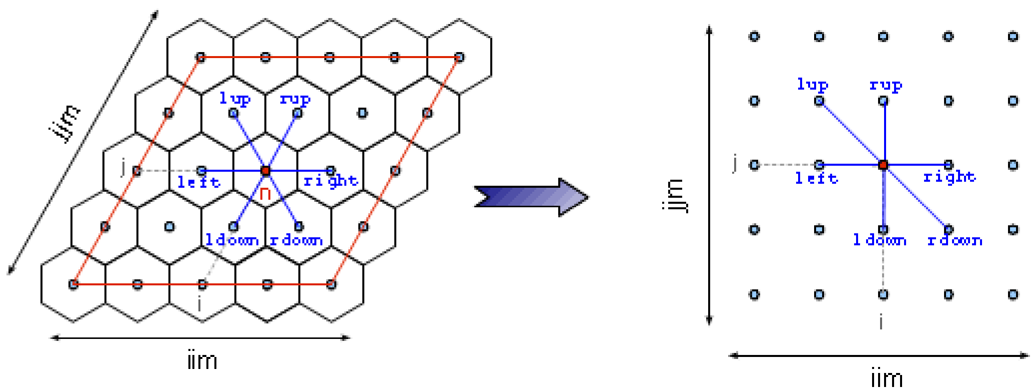


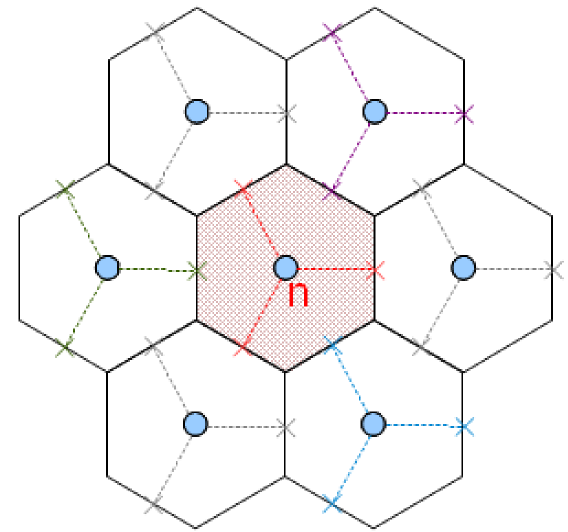


CUDA Hierarchy of threads, blocks, and grids, with corresponding per-thread private, per-block shared, and per-application global memory spaces.

Feasibility for using GPUs for atmospheric models

T. Dubos, Y. Meurdesoif





- Data stored in rectangular arrays
- Direct access to neighbours via constant offsets
- No special case for pentagons (handled by metrics)

```
DO j=jj_begin,jj_end
  DO i=ii_begin,ii_end
    n=(j-1)*iim+i
    dhi(n)=-1./Ai(n)*(ne(n,right)*ue(n+u_right)*le(n+u_right) + &
                      ne(n,rup)*ue(n+u_rup)*le(n+u_rup) +       &
                      ne(n,lup)*ue(n+u_lup)*le(n+u_lup) +       &
                      ne(n,left)*ue(n+u_left)*le(n+u_left) +    &
                      ne(n,ldown)*ue(n+u_ldown)*le(n+u_ldown) + &
                      ne(n,rdown)*ue(n+u_rdown)*le(n+u_rdown))
  ENDDO
ENDDO
```

Feasibility for using GPUs for atmospheric models
T. Dubos, Y. Meurdesoif

Computing with GPUs : programming models

Low-level
- CUDA, CUDA Fortran
- Explicit access to hardware features
- Many optimization opportunities
- Specialized language
- Rapid obsolescence of fine-tuned code due evolving hardware

High-level
- PGI Accelerator, HMPP
- Insert directives in normal Fortran
- Let compiler exploit GPU or similar hardware

How much performance can a low-level approach extract ?
How much of this is lost when using a higher-level approach ?
Which programming patterns can help high-level approaches to perform well ?

2012 : Shallow-water model - low-level vs high-level implementation, tuning and benchmarking

2013 : 3D hydrostatic core

2014 : Introduce (simplified) physics ?

Massively parallel multigrid solvers
J. Thuburn

The scientific case

- All models in ICOMEX currently use some form of explicit or split explicit time integration schemes (perhaps with some vertical terms treated implicitly)
- These are complicated and typically require several ad-hoc damping mechanisms to obtain stability
- Implicit or semi-implicit time integration can have better stability properties and allow much longer time steps...
- ...but require the solution of a global elliptic problem.

Can we do this efficiently and scalably?

Massively parallel multigrid solvers
J. Thuburn

$$\nabla^2\Phi - \frac{1}{L^2}\Phi = R$$

Geometric multigrid:
iterative, well-understood, optimal solution of elliptic problems

Built around V-cycle = sequence of related elliptic problems posed on grids of increasing, then decreasing mesh size

If massively parallel, there may not be enough work to share on the coarsest grid => limit to scalability

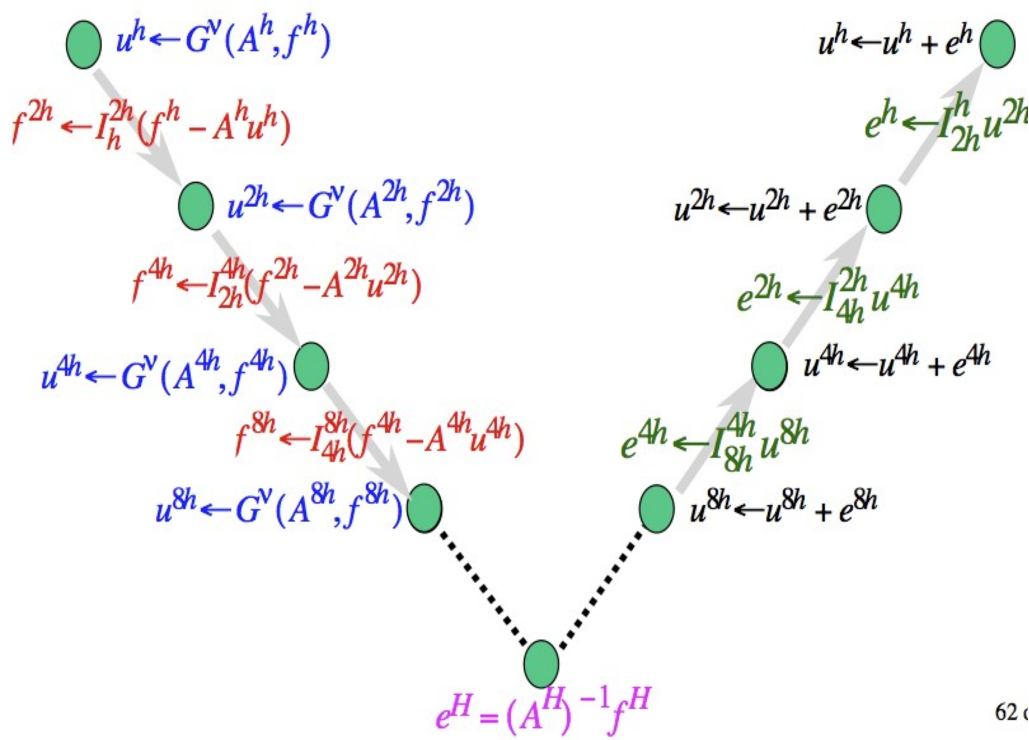
If the elliptic problem comes from semi-implicit time integration*, then*

L~c.dt

where c is the fastest wave phase speed

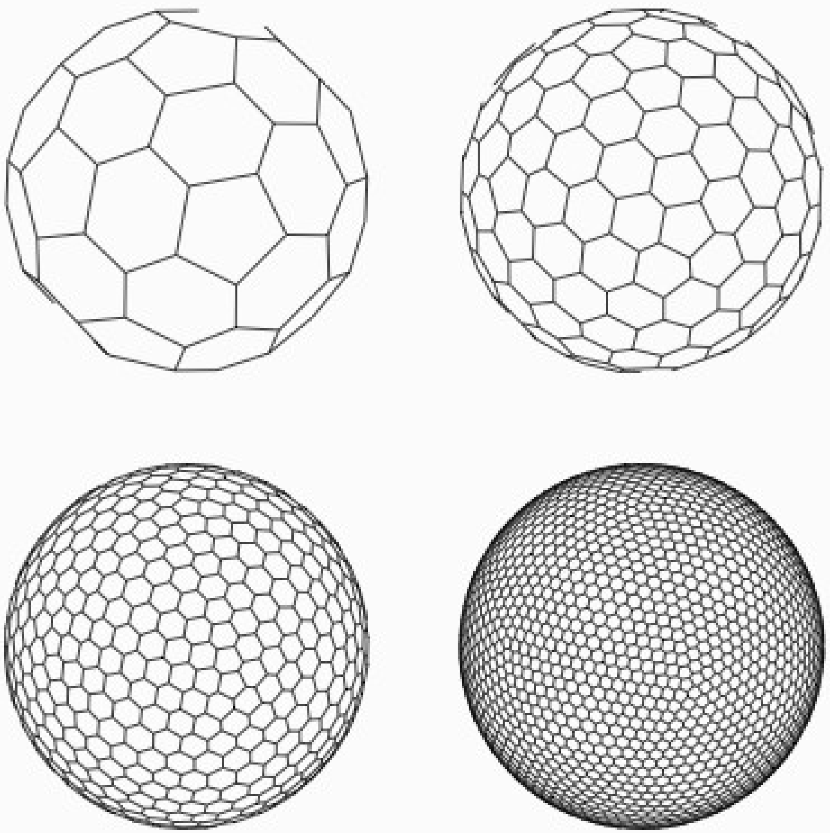
The V-cycle can stop when resolution ~ L without loss of efficiency

=> possibly enough work to share !



Massively parallel multigrid solvers
J. Thuburn

- Standard icosahedral grids provide a natural hierarchy for geometric multigrid (but what about locally refined grids?)



Parallel post-processing server
J. Thuburn, T. Dubos

Why do we need on-line post-processing ?

- Scientific work will typically not use native model grids but more user-friendly grids (lat-lon)
- In many cases fine-scale detail is not analyzed, only temporal /spatial statistics
- The correct interpolation/averaging operations depend on the model numerics
- Part of the post-processing needs to be developed along the model
- Then, why not do (some of) it on-line ?

Potential benefits :

- Reduce the I/O bottleneck, during and/or after the model runs

Approach :

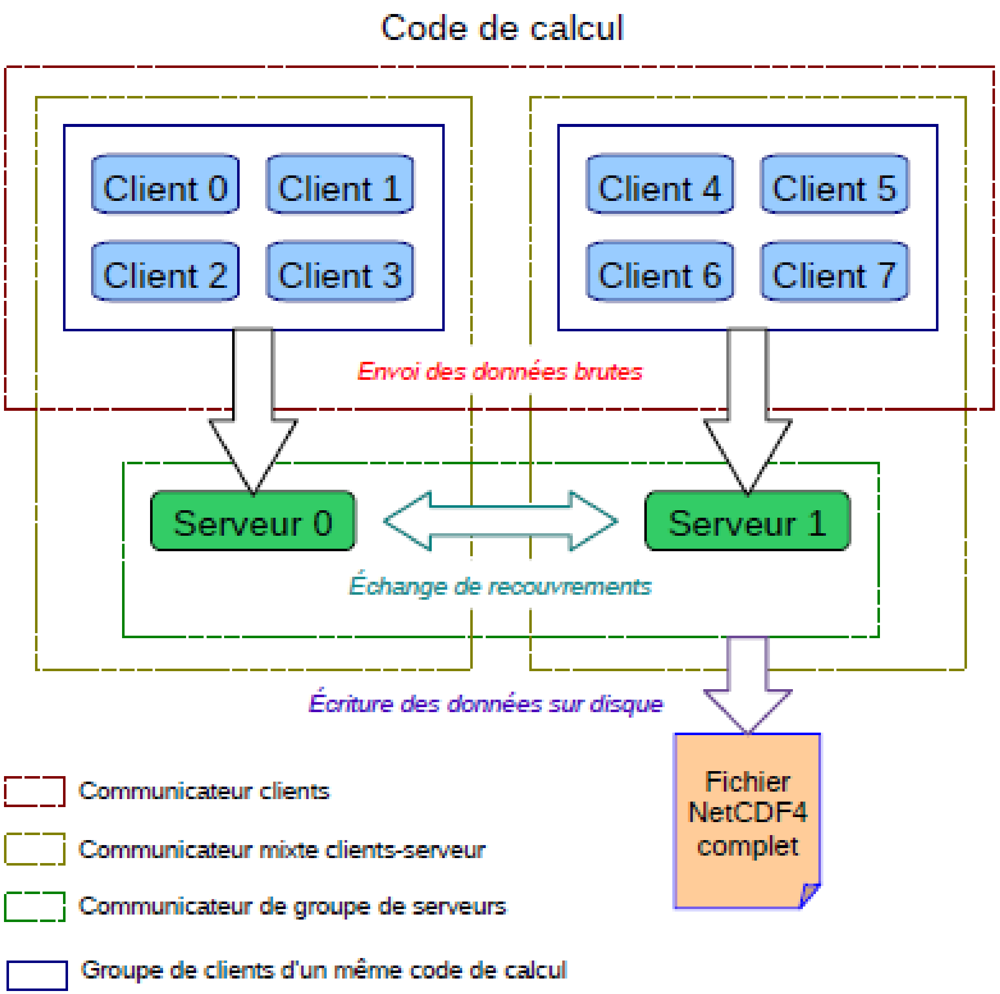
- Extend XML-I/O server (XIOS) to icosahedral grids
- Develop adequate, scalable horizontal/vertical interpolation/averaging operations
- XIOS already provides a parallel framework, high-level description of outputs and basic temporal statistics

Parallel post-processing server
J. Thuburn, T. Dubos

Parallelism in XIOS

- *Client/server approach*
  Non-blocking call : computation can proceed immediately

- *Parallel NetCDF*
  All I/O servers access a single file, removing the need to for post-simulation « rebuild » (merging of per-process files).



Parallel post-processing server
J. Thuburn, T. Dubos

XML description of outputs

- Associates *properties* to items
- Concept of *inheritance*

```
<field id="toce" long_name="temperature (Celcius)" unit="degC" grid_ref="Grid_T" />
<field id="toce_K"  field_ref="toce" long_name="temperature (Kelvin)"  unit="degK" />
```

- *Hierarchical* organization

```
<field_definition/>
  <field_group id="grid_T" domain_ref="grid_T">
    <field id="toce"  long_name="temperature" unit="degC" axis_ref="deptht"   />
    <field id="soce"  long_name="salinity"    unit="psu"  axis_ref="deptht"   />
    <field id="sst"   long_name="sea surface temperature" unit="degC"         />
    <field id="sst2"  long_name="square of sea surface temperature"  unit="degC2" />
    <field id="|sstgrad|"  long_name="module of sst gradient" unit="degC/m" />
  </field_group>
</field_definition>

<file_definition>
   <file id="1d" name="out_1day" output_freq="1day" enabled=".TRUE." />
     <field_group field_group_ref="grid_T" />
   </file>
</file definition>
```

- Avoids repetition
- Compact, flexible and human-readable

Parallel I/O
T. Ludwig

•I/O expected to become a major bottleneck for exascale computing

Even if model uses parallel I/O :

- Bottlenecks main remain in the low-level libraries (e.g. NetCDF, MPI-I/O) and file system
- Usage of low-level libraries by models may be suboptimal

Plan :

- Analyze the current temporal and spatial I/O access patterns of the application layer, and the resulting low level accesses performed by the intermediate I/O libraries.
- Create a benchmark resembling typical I/O access patterns of the model. This benchmark should scale to exaflop systems.
- Benchmark currently deployed file systems in the consortiums compute facilities
- Determine the bottlenecks in hardware or software environment.

- Evaluate different access patterns by using NetCDF, HDF5 and Grib, and compare the peak performance with the performance obtained by the model I/O benchmark. This will reveal optimization potential of current model access patterns.
- Compare performance of the I/O library to peak performance of parallel file system to reveal whether the intermediate I/O library is capable to extract raw performance from the file systems.
- Feedback promising optimization strategies to vendors / developers (e.g. to the NetCDF community to create a new on-disk file format).

Collaboration with hardware vendors
T. Ludwig

- Experts from Cray, Fujitsu, IBM and NEC are associated to the project
- Participate to project meetings and mailing list
- Gain insight into the structure of computation and I/O performed by our models
- Gain insight into the development process of the models
- Provide guidance for implementation strategies
- Provide hints about possible/impossible future directions of hardware/software
- Specif c time devoted to discussions during project meetings

Questions ?