RÉPUBLIQUE
FRANÇAISE
Liberté
Égalité
Fraternité

ONERA
THE FRENCH AEROSPACE LAB

The CWIPI Library

ExCALIBUR Code Coupling Workshop

Bastien Andrieu, Eric Quémerais

December 3rd 2021

Ce document est la propriété de l'ONERA. Il ne peut être communiqué à des tiers et/ou reproduit sans l'autorisation préalable écrite de l'ONERA, et son contenu ne peut être divulgué.
This document and the information contained herein is proprietary information of ONERA and shall not be disclosed or reproduced without the prior authorization of ONERA.

Outline

CWIPI (Coupling With Interpolation Parallel Interface)

Overview

- Library for coupling computational codes in a massively parallel distributed environment
- Provides tools for defining coupling algorithms and controlling exchanges between codes
- Transfer of interpolated fields through a geometric coupling interface
- LGPL3 License
- Development started in 2009 based on the FVM library (now PLE by Y. Fournier, EDF)
- Multiple call interfaces: C/C++, Fortran, Python/Numpy and Pyhton/CGNS *via* PyC2
- Download at `https://w3.onera.fr/cwipi/bibliotheque-couplage-cwipi`

Contributors

- ONERA
- CERFACS

Contact

eric [dot] quemerais [at] onera [dot] fr | bastien [dot] andrieu [at] onera [dot] fr

Two operating modes

Direct call	Call *via* the OpenPALM* coupler (CERFACS)
The coupling algorithm is defined by the codes. No dedicated coupling process.	The coupling algorithm is defined with an HMI and driven by the OpenPALM process.
MPI communications	MPI and TCP/IP communications
Users: • Safran AE, Safran Tech, ArianeGroup • CERFACS, CORIA, LEGI, Université de Brest, TU Darmstadt	**Users:** • Safran, Siemens, Airbus • CERFACS, EM2C, CORIA, Météo France, University of Surrey, LMFA, UCLouvain, CNES
Instrumented codes: CEDRE, ASTRE, elsA, Marc, SPACE, Zebulon, AVBP, YALES2, Nektar++, ...	**Instrumented codes:** AVBP, AVTP, Marc, SAMCEF, ABAQUS, Fluent, NASTRAN, FINE™/Turbo, ...

* https://www.cerfacs.fr/globc/PALM_WEB/

MPI communications management

Execute multiple parallel codes in a single MPI run

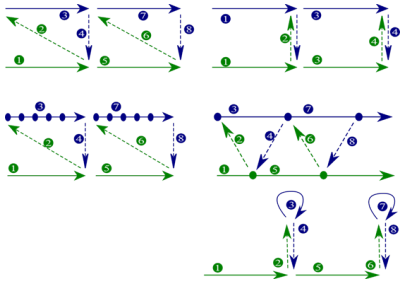
```
mpirun -np <n1> python_code.py : -np <n2> c_code.exe : -np <n3> fortran_code.exe
```

MPI Communicators

- Creation of each code's MPI intra-communicators
- Creation of coupling MPI communicators
  - No limit on the number of couplings

Exchange of global control parameters

- Global, shared variables (`int`, `double`, `char *`, ...)
- Useful tools for designing coupling algorithms in codes
- No default control parameter
- Examples: time, time step, ...

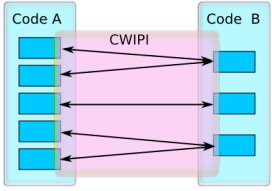


Examples of coupling algorithms (J.D. Garaud)

Geometric coupling interface

Interface type

- Line: segments
- Surface: triangles, quadrangles, non-convex polygons
- Volume: tetrahedra, pyramids, prisms, hexahedra, non-convex polyhedra
- Surface immersed in a volume
- Non-matching



Example of communication graph

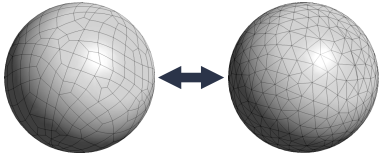
Communications through the coupling interfaces

- Construction of the Point-to-Point communication graph between the processes of both coupled codes
  - resulting from the localization of target points in the partitions of the source mesh
- Exchange of the interpolated fields
  - blocking/non-blocking MPI communications

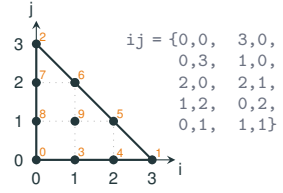
Geometric coupling interface: high-order meshes

High-order, curved elements are supported

- Elements of any geometric order
  - line
  - surface: triangles, quadrangles
  - volume: tetrahedra, pyramids, prisms, hexahedra
- User-defined node numbering inside an element
  - `cwipi_ho_ordering_from_IJK_set`
  - input and output data conform to this numbering
- Advanced use: the geometric order and the order of the numerical method can be dissociated



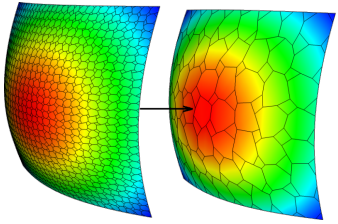
*Q*2/*P*2 coupling



User-defined node numbering in a *P*3 triangle

Standard workflow

Localization of target degrees-of-freedom in the source mesh



Source/Target fields

- Default dof location: nodes or cell centers

Default interpolation

- of the order of the element if the source field is defined at nodes
- value of the containing cell if the source field is defined at cell centers

Exchange interpolated fields

Export interpolated fields for visualization

Basic example: Fortran code (CodeF) coupled with C code (CodeC)

```
call CWIPI_INIT_F &
       (comm, "CodeF", intraComm)
...
call CWIPI_CREATE_COUPLING_F &
       ("cpl1", "CodeC", ...)
...
call CWIPI_DEFINE_MESH_F &
       ("cpl1", mesh, ...)
...
call CWIPI_LOCATE_F("cpl1")
...
call CWIPI_SYNCH_CTRL_PARAM_F &
       ("CodeC")
call CWIPI_GET_DIST_PARAM_F &
       ("CodeC","niter", niter)
...
do i = 1, niter
  call CWIPI_ISSEND_F &
         ("cpl1", "exch1", srcField)
  call CWIPI_WAIT_ISSEND_F &
         ("cpl1", ...)
enddo
...
call CWIPI_DELETE_COUPLING_F("cpl1")
call CWIPI_FINALIZE_F()
```

Initialize MPI communicators

Define the coupling object
object name, coupled code, coupling type, exchanged fields type, geometric tolerance, ...

Define interface meshes

Localize
target dofs in source mesh

Control the coupling
CodeC indicates to CodeF how many exchanges to perform

Exchange fields
send interpolated field from CodeF to CodeC

Delete the coupling object

End communications

```
cwipi_init
  (comm, "CodeC", intraComm);
...
cwipi_create_coupling
  ("cpl1", "CodeF", ...);
...
cwipi_define_mesh
  ("cpl1", mesh,...);
...
cwipi_locate("cpl1");
...
cwipi_add_local_int_control_parameter
  ("niter", 10);
cwipi_synchronize_control_parameter
  ("CodeC");
...
for (int i = 0; i < 10; i++) {
  cwipi_irecv
    ("cpl1", "exch1", tgtField);
  cwipi_wait_irecv
    ("cpl1",...);
}
...
cwipi_delete_coupling("cpl1");
cwipi_finalize();
```

Python interface

Dependencies

- **MPI4PY**: MPI wrapping
- **NumPy**: wrapping of arrays (exchanged fields, coordinates, connectivities, ...)
- **Cython**: wrapping of C API

Example

```
import mpi4py.MPI as MPI
import numpy as np
import cwipi
...
f = open("output")
cwipi.set_output_listing(f)          # Output redirection
cwipi.init(MPI.COMM_WORLD, ...)      # Initialization
cpl = cwipi.Coupling("cpl1", ...)    # Coupling creation
cpl.define_mesh(...)                 # Coupling interface definition
cpl.locate()                         # Geometric localization
cpl.exchange("exch1", ...)           # Exchange
del cpl                              # Coupling destruction
cwipi.finalize()                     # Finalization
```

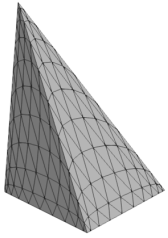
Advanced use

User-defined target points

- Gauss quadrature points, ...

User-defined interpolation method *via* callbacks

- Input data provided by CWIPI
  - For each target point located on the current process:
    - *xyz*-coordinates
    - containing or closest element
    - parametric coordinates in that element
    - distance from that element
  - Pointer to the source field
    - from call to `cwipi_send`/`cwipi_issend`
    - free format: data used only in this function
- Output: field interpolated at target points
- Access to the internal structures of the coupled code
- Calls to the basis functions and internal interpolation methods of the coupled code

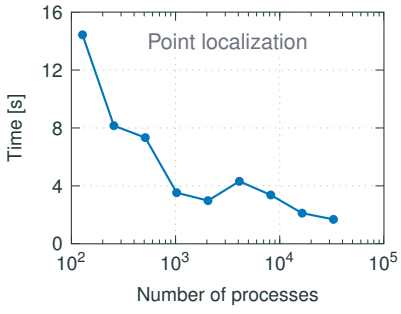


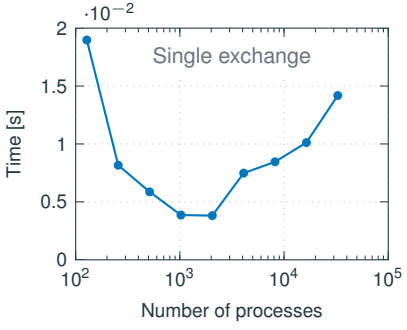
Optimized $P10$ tetrahedron (C. Peyret)

High-performance code coupling

AVBP/AVBP coupling (F. Duchaine CERFACS, 2015)

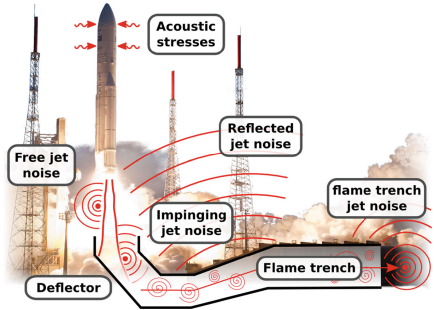
- Turbomachine simulation with a volume coupling interface
- 2×32,000 cores on Titan (Oak Ridge National Laboratory)
- Each instance of AVBP treats 400M elements
- Coupling interface partitioned on 4,000 + 5,000 cores





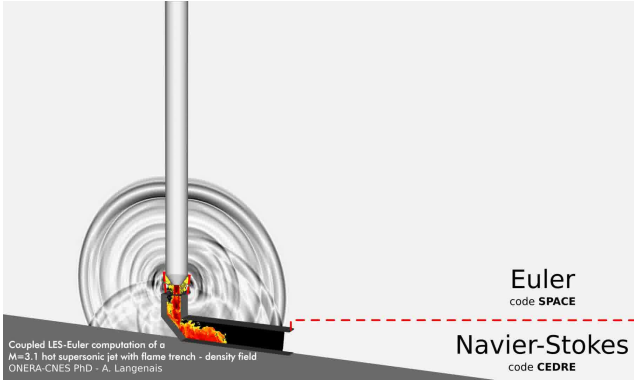
Outline

Application: Noise generated by a rocket engine at lift-off



A. Langenais, F. Vuillot, C. Peyret, G. Chaineray (ONERA), C. Bailly (LMFA)

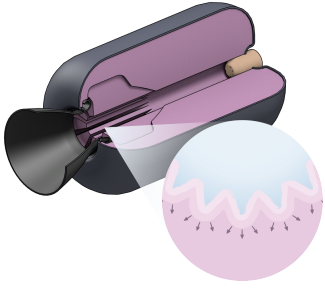
[Flow Turbulence and Combustion, 2018]

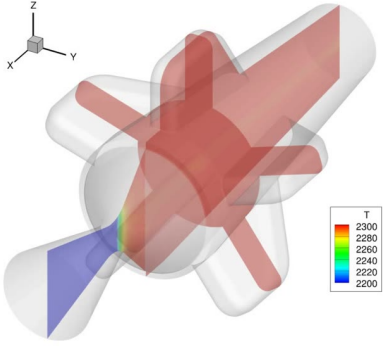


Application: Aero-Mechanical coupling

Interaction between fluid flow and grain regression

- Computation with CEDRE
- N. Lupoglazoff, G. Chaineray (ONERA)

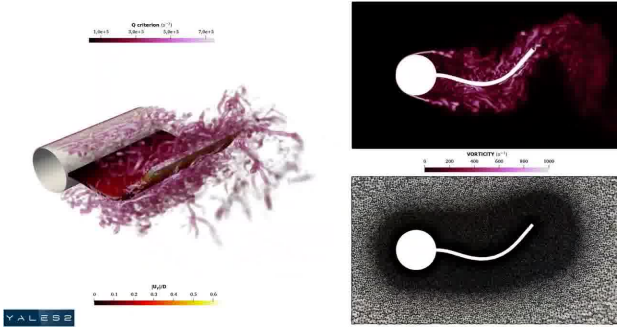




Application: Fluid-Structure Interaction

Turbulent flow around a fixed cylinder with a flexible thin rubber plate

- Coupling of an ALE solver with LES approach and a Structural Mechanics Solver (YALES2)
- T. Fabbri, G. Balarac (LEGI), P. Bénard, V. Moureau (CORIA) (2021)

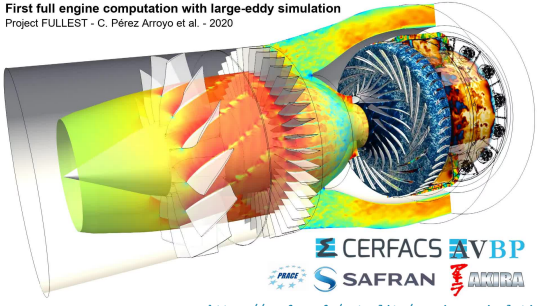


https://www.youtube.com/watch?v=X7Zb7y67yAo

Application: Large-Eddy Simulation of a full aircraft engine

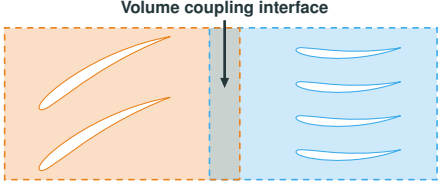
3 instances of AVBP

- 2 billion elements
- C. Pérez Arroyo, J. Dombard, F. Duchaine, L. Gicquel, B. Martin, N. Odier, G. Staffelbach (CERFACS) [JGPPS, 2020]



2 coupling instances of CWIPI

- Moving, *volume* coupling interfaces
  - Localization performed at each exchange
  - Results too large to be stored
  - Efficient algorithms are mandatory

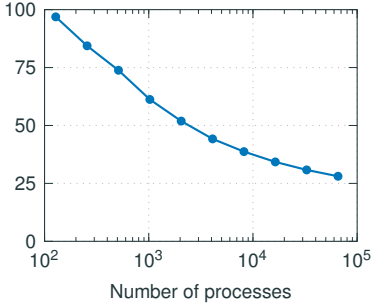


https://cerfacs.fr/actualite/premiere-simulation-haute-fidelite-360-degres-dun-moteur-complet/

Outline

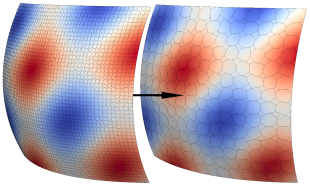
Rewriting of the geometric localization algorithm

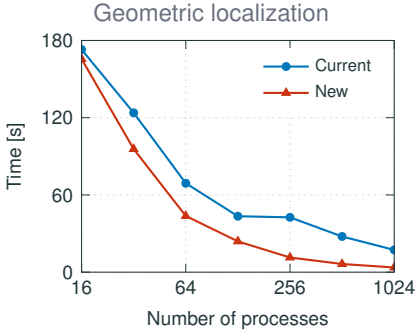
- Arbitrary repartition of the partitions of both source and target meshes on processes
  - each process can send and receive data
- Use of a parallel, distributed octree structure
- Dynamic load balancing
  - *all* processes are involved

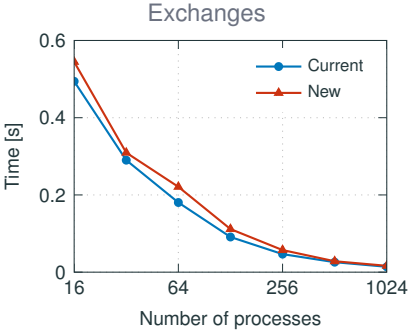


Percentage of processes with a part of the coupling interface in a turbomachine calculation (F. Duchaine, CERFACS)

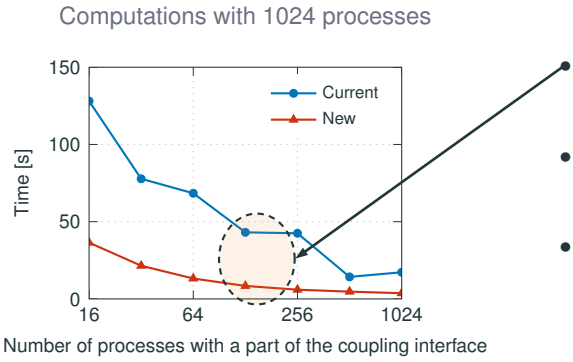
Rewriting of the geometric localization algorithm







Influence of dynamic load balancing

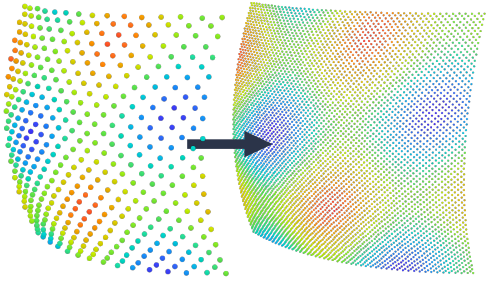


- Typical range of data distribution in calculations using CWIPI
- An even greater improvement is expected for volume and high-order coupling interfaces
- The cost of load balancing is minor compared to geometric computations

New spatial interpolation methods (1/2)

Interpolation in a point cloud

- Parallel *k*-NN algorithm to find closest source points
- Least-square interpolation by default
- User-defined interpolation methods *via* callbacks
  - Radial Basis Functions, . . .
- Performance on 960 Cascade Lake processors:
  - 100M$\times$100M points, $k = 5$
  - Total elapsed time: **3.85 s**
    - *k*-NN search: 3.75 s
    - interpolation and transfer: 0.10 s



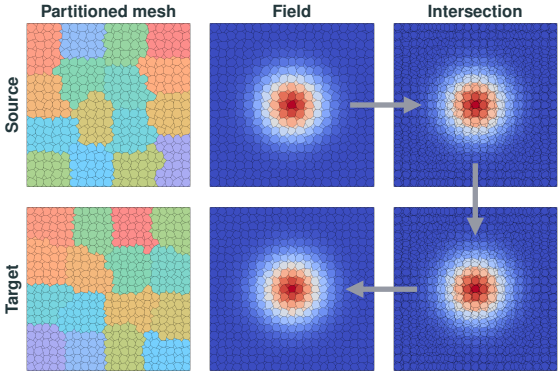
New spatial interpolation methods (2/2)

Conservative interpolation

- Intersection of source and target meshes
  - 2D & Surface: 2022
  - Volume: 2023

Performance on 1024 Broadwell processors

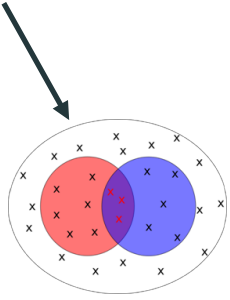
	3M×3M faces	300M×300M faces
Total elapsed time	**3.76 s**	**447 s**
Build bounding-box tree	0.07 s	2 s
Search candidates	1.43 s	54.0 s
Balance intersections	0.15 s	11 s
Compute intersections	1.57 s	307 s
Build connectivities	0.5 s	23 s
Other	0.07 s	50 s



Roadmap

Major new version (expected mid 2022)

- Arbitrary distribution of the partitions of the coupled codes on processes
- Access to multiple types of spatial interpolation methods
- Non-blocking synchronization of control parameters using MPI RMA



Asynchronous coupling (2023)

- Time interpolation
- Efficient storage of fields

Client-Server mode (2023)

HPC

- MPI/GPU hybridation (2023)
- MPI RMA (2023)

Thank you for your attention

www.onera.fr