

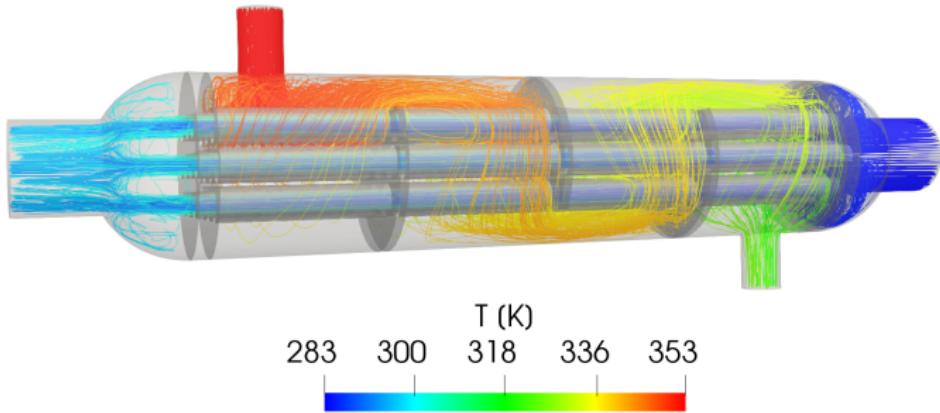


A Coupling Library
for Partitioned Multi-Physics
Simulations

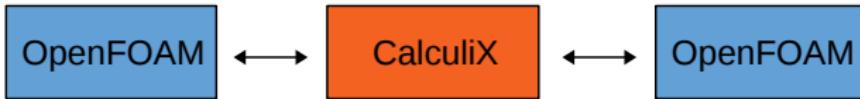
- 1.** What is preCICE?
- 2.** How to get started?
- 3.** How can I couple my own code?

1. What is preCICE?

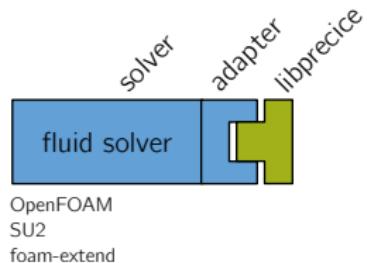
Example: Shell-And-Tube Heat Exchanger



- ▶ Partitioned coupling: Usage of three independent solvers
- ▶ Reuse of existing solvers



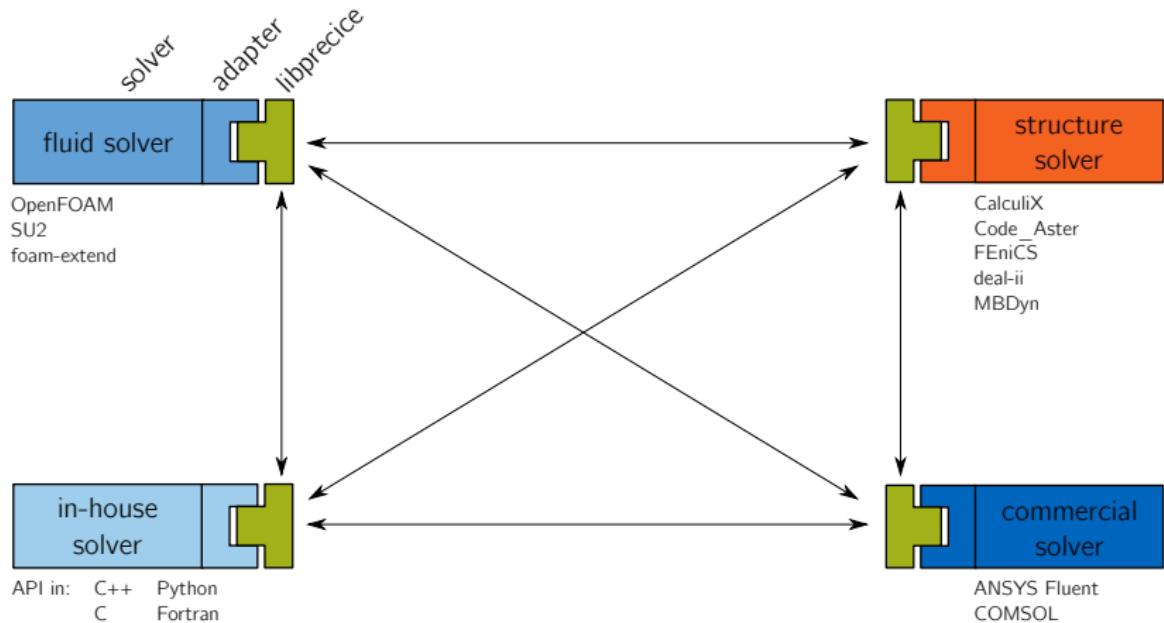
preCICE – A Plug-and-Play Coupling Library



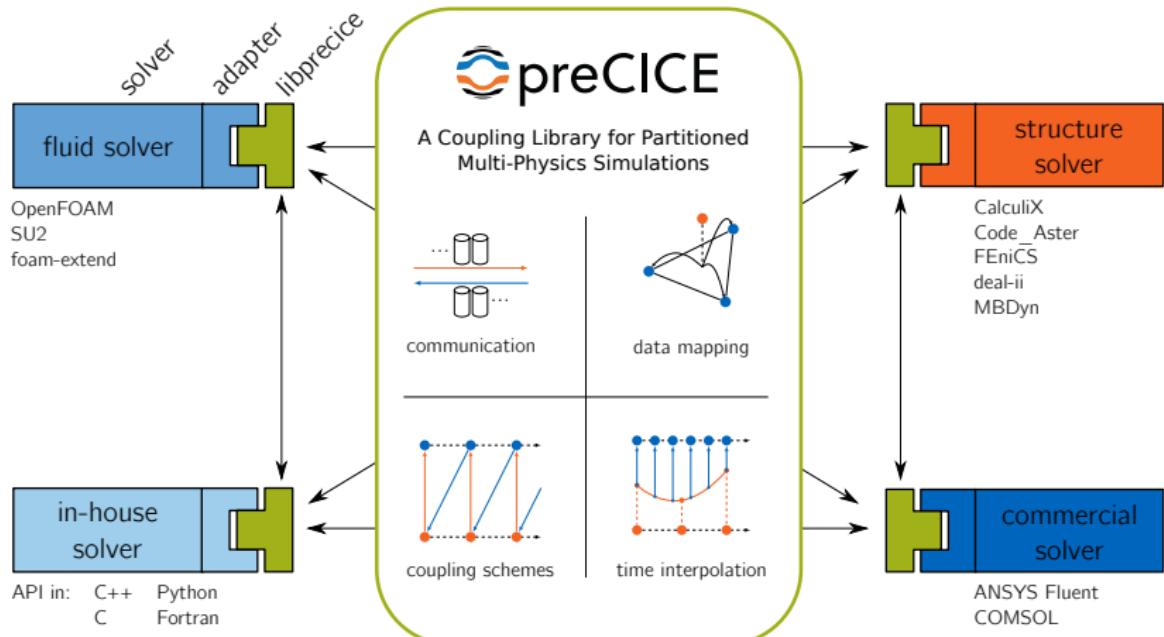
preCICE – A Plug-and-Play Coupling Library



preCICE – A Plug-and-Play Coupling Library



preCICE – A Plug-and-Play Coupling Library

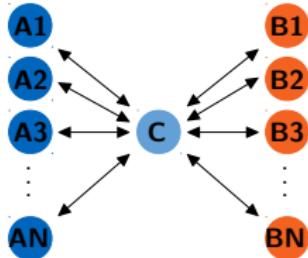


Unique Selling Points (USPs)

1. Scalability
2. Robust quasi-Newton coupling
3. Coupling of arbitrary many components
(arbitrary many = more than two)
4. Minimally-invasive coupling
5. Open-source, community

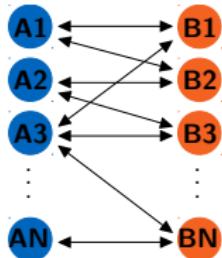
USP 1: Scalability

Server-Based Concept



- ▶ Complete communication through central server process
- ▶ Interface computations on server (in sequential)
- ▶ ⇒ Coupling becomes bottleneck for overall simulation already on moderate parallel systems

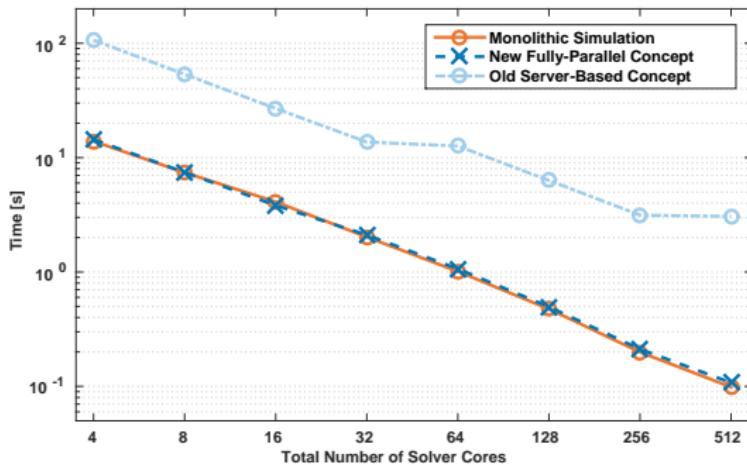
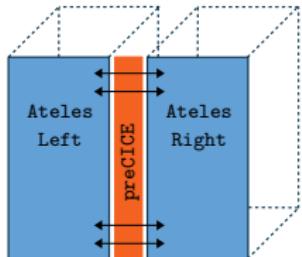
Our Peer-To-Peer Concept



- ▶ No central entity
- ▶ ⇒ Easier to handle (user does not need to care about server)
- ▶ ⇒ No scaling issues

USP 1: Scalability

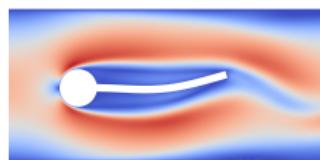
- ▶ Travelling density pulse (Euler equations) through artificial coupling interface
- ▶ DG solver Ateles (U Siegen), $7.1 \cdot 10^6$ dofs
- ▶ Nearest neighbor mapping and communication



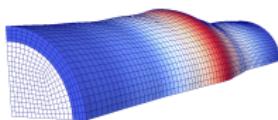
USP 2: Quasi-Newton Coupling

Coupled problem: $F : d \mapsto f$, $S : f \mapsto d \rightsquigarrow (S \circ F)(d) \stackrel{!}{=} d$

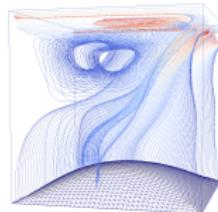
FSI3



3D-Tube



Driven Cavity



Mean Iterations	Aitken	Quasi-Newton
FSI3	17.0	3.3
3D-Tube	Div.	7.5
Driven Cavity	7.4	2.0

USP 2: Quasi-Newton Coupling

- ▶ Quasi-Newton can even handle biomedical applications, such as an Aortic bloodflow
- ▶ Stable coupling (no added-mass instabilities)
- ▶ Six times less iterations than Aitken



-
- ▶ Joint work with Juan-Carlos Cajas (Barcelona Supercomputing Center)
 - ▶ Geometry by Jordi Martorell

Contributors



Miriam Mehl
U Stuttgart



Florian Lindner
U Stuttgart



Amin
Totounferoush
U Stuttgart



Kyle Davis
U Stuttgart



Alexander Rusch
ETH Zürich



Hans Bungartz
TUM



Benjamin Rüth
TUM



Gerasimos
Chourdakis
TUM



Frédéric Simonis
TUM



Benjamin
Uekermann
TU/e

Previous and minor contributors:

- ▶ Bernhard Gatzhammer, Klaudius Scheufele, Lucia Cheung, Alexander Shukaev, Peter Vollmer, Georg Abrams, Alex Trujillo, Dmytro Sashko, David Sommer, David Schneider, Richard Hertrich, Saumitra Joshi, Peter Meisrimel, Derek Risseeuw, Rafal Kulaga, Ishaan Desai ...

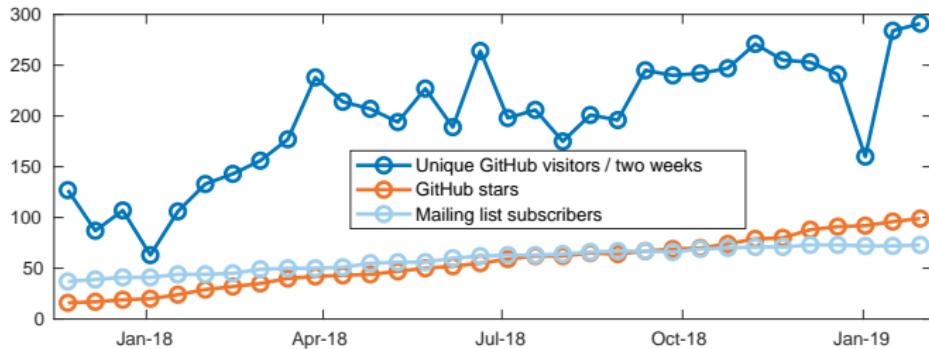
Users

- ▶ LSM & STS, U Siegen, Germany
- ▶ SC & FNB, TU Darmstadt, Germany
- ▶ SCpA, CIRA, Italy
- ▶ Cardiothoracic Surgery, UFS, South Africa
- ▶ A*STAR, Singapore
- ▶ NRG, Petten, The Netherlands
- ▶ Aerodynamics & Wind Energy (KITE Power), TU Delft, The Netherlands
- ▶ Mechanical and Aeronautical Eng., University of Manchester, UK
- ▶ University of Strathclyde, Glasgow, UK
- ▶ FAST, KIT, Germany
- ▶ AIT, Vienna, Austria

- ▶ IAG, University of Stuttgart, Germany
- ▶ CTTC UPC, Barcelona, Spain
- ▶ Amirkabir U. of Technology, Iran

Upcoming:

- ▶ GRS, Garching, Germany
- ▶ MTU Aero Engines, Munich, Germany
- ▶ Numerical Analysis, Lund, Sweden
- ▶ Helicopter Technology & Astronautics, TUM, Germany
- ▶ ATA Engineering Inc., USA
- ▶ BITS Pilani, India
- ▶ Aviation, MSU Denver, USA



2. How to get started?

Infrastructure

We are on GitHub: <https://github.com/precice>

The screenshot shows the GitHub repository page for the preCICE project. At the top, there's a search bar and navigation links for Pull requests, Issues, Marketplace, and Explore. Below the header is the project logo, which is a stylized orange and blue circular pattern, followed by the text "preCICE" and a subtitle "A Coupling Library for Partitioned Multi-Physics Simulations on Massively Parallel Systems". A pinned repository card for "precice" is displayed, showing its description, language (C++), stars (52), and forks (17). The main repository card for "precice" follows, featuring its name, description, tags (Simulation, high-performance-computing, openfoam, multiphysics), and stats (52 stars, 17 forks, 1 issue, updated 5 days ago). To the right, there are sections for Top languages (C++, C, Python, HTML) and Most used topics (fluid-structure-interaction, precice, co-simulation, multi-physics, conjugate-heat-transfer). At the bottom, there's a People section showing profile icons for several contributors.

Building

Dependencies

- ▶ Eigen, Boost (version \geq 1.65), libxml2
- ▶ Optional: PETSc, Python (incl. Numpy), MPI

The easy way

- ▶ Ubuntu 18.04: All dependencies available through distribution
- ▶ Ubuntu 16.04: All dependencies available except Boost

Still doable

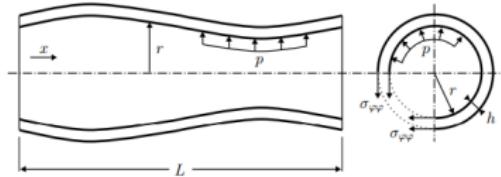
- ▶ macOS
- ▶ Other Linux distributions

Experimental

- ▶ Conda, Docker, Debian package, Spack
- ▶ Windows

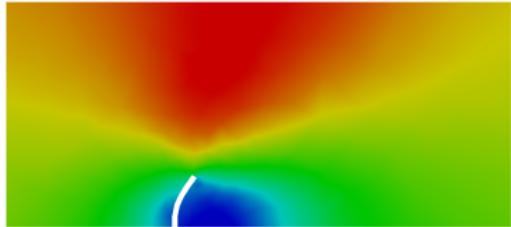
1D Elastic Tube

- ▶ Simple provided solvers
- ▶ Learn about API and configuration



Flexible beam

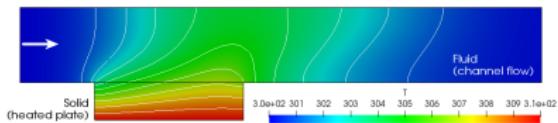
- ▶ Fluid-structure interaction
- ▶ Couple SU2 or OpenFOAM to CalculiX
- ▶ Learn about coupling schemes
- ▶ Also interactive version available in browser <http://run.coplon.de/>



Tutorials

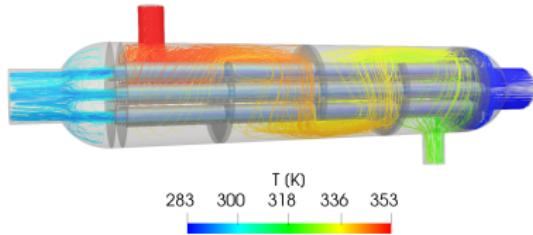
Flow over a Heated Plate

- ▶ Conjugate-heat transfer
- ▶ Couple two OpenFOAM solvers
- ▶ Learn about OpenFOAM adapter



Heat exchanger

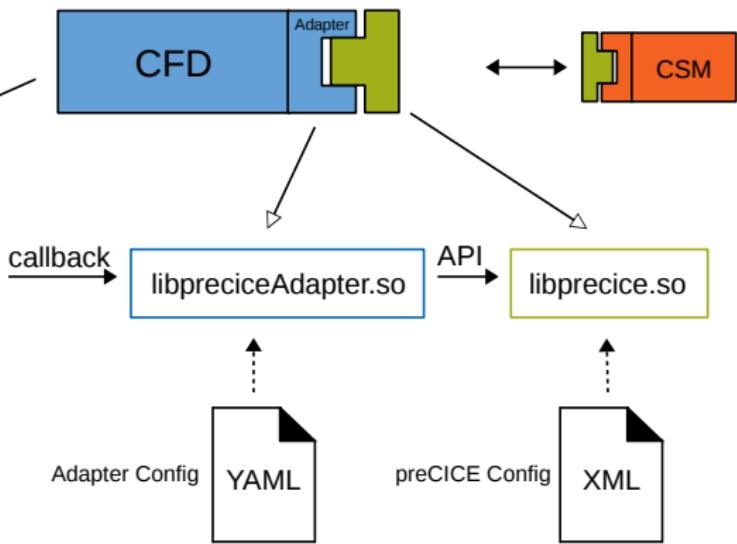
- ▶ Conjugate-heat transfer
- ▶ Couple two OpenFOAM instances with CalculiX
- ▶ Learn about multi coupling



The OpenFOAM Adapter

```
int main(int argc, char *argv[])
{
    #include "setFoamLibPaths.h"
    #include "createTime"
    #include "createMesh.h"
    ...
    while (runTime.loop())
    {
        Info<< "Time = " << endl;
        #include "CourantNo.H"
        // Momentum predictor
        feVectorMatrix UEqn
        (
            fvm::ddt(U)
            + fvm::div(phi, U)
            - fvm::laplacian(nu, U)
        );
    }
}
```

myFoam



Flow over a Heated Plate

Load adapter at runtime in system/controlDict:

```
1 functions
2 {
3     preCICE_Adapter
4     {
5         type preciceAdapterFunctionObject;
6         libs ("libpreciceAdapterFunctionObject.so");
7     }
8 }
```

Define coupling boundary in system/blockMeshDict:

```
1 interface
2 {
3     type wall;
4     faces
5     (
6         (4 0 1 5)
7     );
8 }
```

Flow over a Heated Plate

Configure adapter in precice-adapter-config.yml:

```
1 participant: Fluid
2
3 precice-config-file: /path/to/precice-config.xml
4
5 interfaces:
6 - mesh: Fluid-Mesh
7   patches: [interface]
8   write-data: Temperature
9   read-data: Heat-Flux
```

Flow over a Heated Plate

```
/bin/bash 83x50
[preciceAdapter] [DEBUG]     write-data :
[preciceAdapter] [DEBUG]       HeatFlux
[preciceAdapter] [DEBUG]     read-data :
[preciceAdapter] [DEBUG]       Temperature
[preciceAdapter] [DEBUG]     subcycling : 1
[preciceAdapter] [DEBUG]     prevent early exit : 1
[preciceAdapter] [DEBUG]     evaluate boundaries : 1
[preciceAdapter] [DEBUG]     disable checkpointing : 0
[preciceAdapter] [DEBUG]     CHT mode enabled : 1
[preciceAdapter] [DEBUG] Configuring the CHT module...
[preciceAdapter] [DEBUG]     user-defined solver type : none
[preciceAdapter] [DEBUG]     temperature field name : T
[preciceAdapter] [DEBUG]     transportProperties name : transportProperties
[preciceAdapter] [DEBUG]     conductivity name for basic solvers : k
[preciceAdapter] [DEBUG]     density name for incompressible solvers : rho
[preciceAdapter] [DEBUG]     heat capacity name for incompressible solvers : Cp
[preciceAdapter] [DEBUG]     Prandtl number name for incompressible solvers : Pr
[preciceAdapter] [DEBUG]     turbulent thermal diffusivity field name for incomp
possible solvers : alphaT
[preciceAdapter] [DEBUG] Determining the solver type...
[preciceAdapter] [DEBUG] Found the transportProperties dictionary.
[preciceAdapter] [DEBUG] Did not find the turbulenceProperties dictionary.
[preciceAdapter] [DEBUG] Did not find the thermophysicalProperties dictionary.
[preciceAdapter] [DEBUG] This is a basic solver, as transport properties are provided, while turbulence or transport properties are not provided.
[preciceAdapter] [DEBUG] Checking the timestep type (fixed vs adjustable)...
[preciceAdapter] [DEBUG] Timestep type: fixed.
[preciceAdapter] [DEBUG] Creating the preICE solver interface...
[preciceAdapter] [DEBUG] Number of processes: 1
[preciceAdapter] [DEBUG] MPI rank: 0
[preciceAdapter] [DEBUG] preICE solver interface was created.
[preciceAdapter] [DEBUG] Configuring preICE...
(0) 16:10:32 [impl::SolverInterfaceImpl]:119 in configure: Configuring preICE with configuration: "precice-config.xml"
(0) 16:10:32 [impl::SolverInterfaceImpl]:153 in configure: Run in coupling mode
[preciceAdapter] [DEBUG] preICE was configured.
[preciceAdapter] [DEBUG] Creating interfaces...
[preciceAdapter] [DEBUG] Interface created on meshSolid-Mesh
[preciceAdapter] [DEBUG] Adding coupling data writers...
[preciceAdapter] [DEBUG] Constructed KappaEff_Basic.
[preciceAdapter] [DEBUG] Name of transportProperties: transportProperties
[preciceAdapter] [DEBUG] Name of conductivity: k
[preciceAdapter] [DEBUG] k = 100_000000
[preciceAdapter] [DEBUG] Added writer: Heat Flux for basic solvers.
[preciceAdapter] [DEBUG] Adding coupling data readers...
[preciceAdapter] [DEBUG] Added reader: Temperature.
[preciceAdapter] [DEBUG] Initializing the preICE solver interface...
(0) 16:10:32 [impl::SolverInterfaceImpl]:216 in initialize: Setting up master communication to coupling partner/s
] ]
```

Solid: waiting...

```
/bin/bash 83x50
Create mesh for time = 0
PIMPLE: Operating solver in PISO mode
Reading thermophysical properties
Selecting thermodynamics package
(
    type          heRhoThermo;
    mixture       pureMixture;
    transport     const;
    thermo        hConst;
    equationOfState perfectGas;
    specie        specie;
    energy         sensibleEnthalpy;
)
Reading field U
Reading/calculating face flux field phi
Creating turbulence model
Selecting turbulence model type laminar
Selecting laminar stress model Stokes
Reading g
Reading hRef
Calculating field g.h
Reading field p_rgh
Creating field dpdt
Creating field kinetic energy K
No MRF models present
Radiation model not active: radiationProperties not found
Selecting radiationModel none
No finite volume options present
Courant Number mean: 0.0837143 max: 0.403668
Starting time loop
] ]
```

Fluid: preparing...

3. How can I couple my own code?

How to couple my own code?

```
1     precice::SolverInterface precice("FluidSolver",rank,size);
2     precice.configure("precice-config.xml");
3     precice.setMeshVertices();
4     precice.initialize();
5
6     while (precice.isCouplingOngoing()) { // main time loop
7         solve();
8
9         precice.writeBlockVectorData();
10        precice.advance();
11        precice.readBlockVectorData();
12
13        endTimeStep(); // e.g. write results, increase time
14    }
15
16    precice.finalize();
```

- ▶ Timesteps, most arguments, and less important methods omitted.
- ▶ Full example in the wiki.
- ▶ API in C++, C, Fortran, and Python

Roadmap

Current Developments v1.4

- ▶ Debian package
- ▶ Building with cmake

Current Developments Adapters

- ▶ Fluid-fluid module for the OpenFOAM adapter
- ▶ Update of Fluent adapter
- ▶ Official adapters for dealii, FEniCS, and Nutils

Long-term Goals v2.0

- ▶ 3D-1D and 3D-2D data mapping
- ▶ Parallel initialization → support of very large cases
- ▶ Support of re-meshing and dynamically changing coupling interfaces
- ▶ Consistent time interpolation

Funding



TUM TU/e





Flexible: Couple your own solver with any other

Easy: Add a few lines to your code

Ready: Out-of-the box support for many solvers

Fast: Fully parallel, peer-to-peer, designed for HPC

Stable: Implicit coupling, accelerated with Quasi-Newton

Multi-coupling: Couple more than two solvers

Free: LGPL3, source on GitHub

- www.precice.org
- github.com/precice
- @precice.org
- Mailing-list, Gitter
- Literature Guide on wiki

