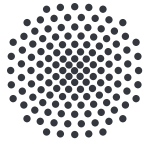
SIAM CSE25 - Fort Worth, TX, USA - March 5, 2025

Black-box coupling of simulation codes using preCICE

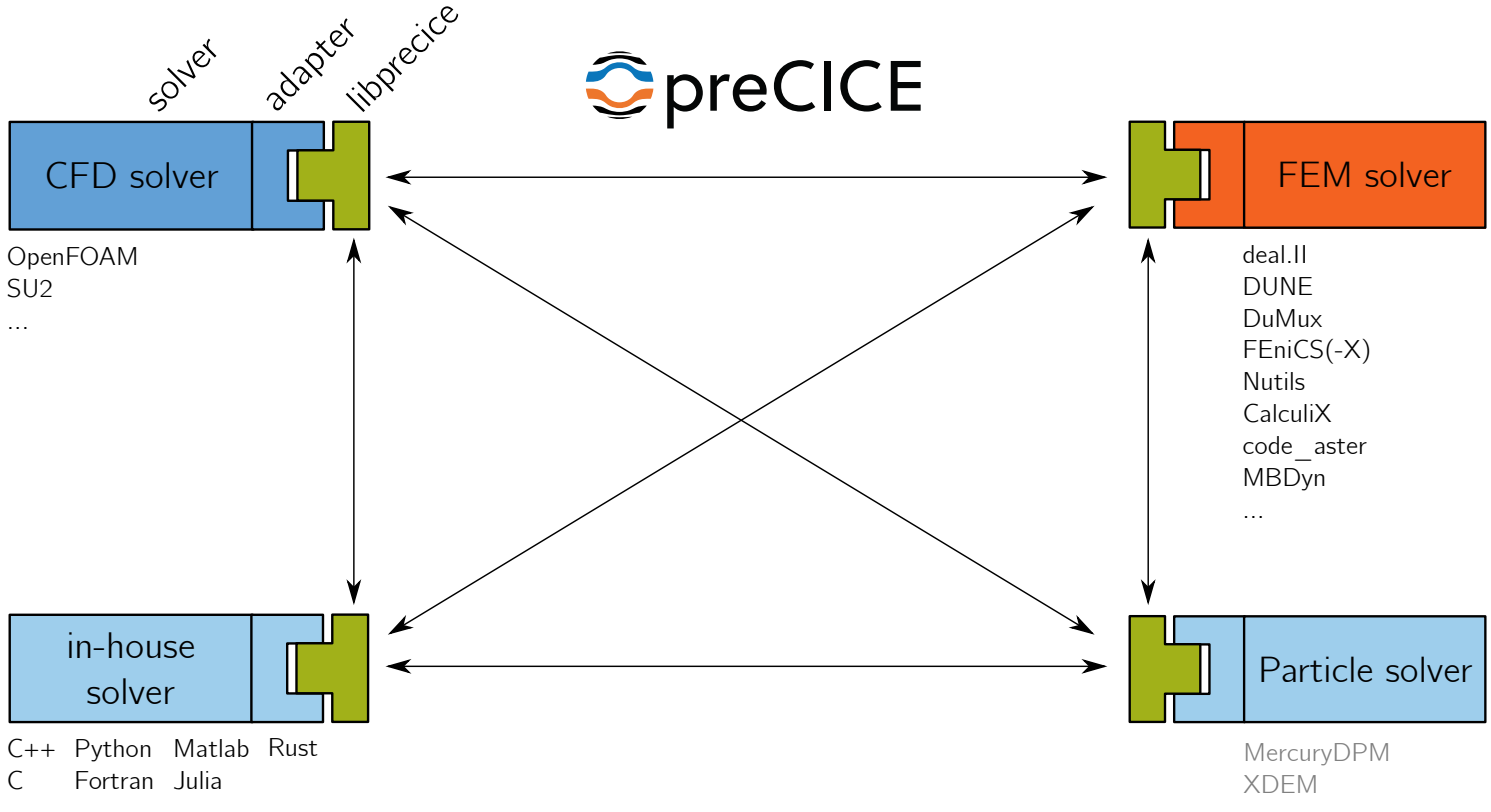
Gerasimos Chourdakis

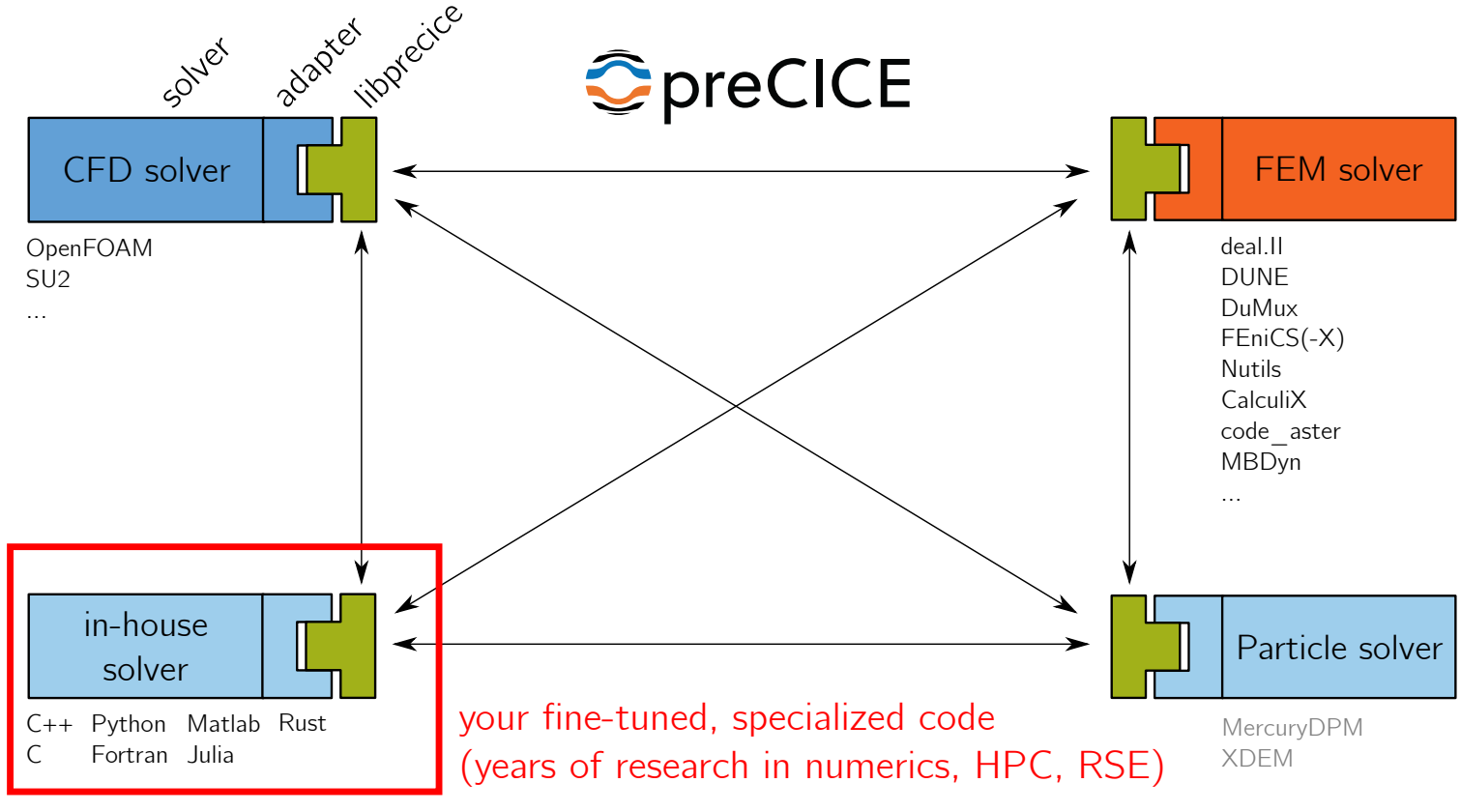
Institute for Parallel and Distributed Systems
University of Stuttgart
gerasimos.chourdakis@ipvs.uni-stuttgart.de



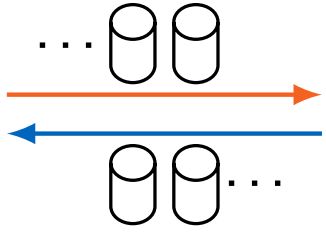
University of Stuttgart
Germany


solver
adapter
libprecice
preCICE
CFD solver
FEM solver


solver
adapter
libprecice
preCICE
CFD solver
OpenFOAM
SU2
...
FEM solver
deal.II
DUNE
DuMux
FEniCS(-X)
Nutils
CalculiX
code_aster
MBDyn
...
in-house solver
C++ Python Matlab Rust
C Fortran Julia
Particle solver
MercuryDPM
XDEM


solver
adapter
libprecice
preCICE
CFD solver
OpenFOAM
SU2
...
FEM solver
deal.II
DUNE
DuMux
FEniCS(-X)
Nutils
CalculiX
code_aster
MBDyn
...
in-house solver
C++ Python Matlab Rust
C Fortran Julia
Particle solver
MercuryDPM
XDEM
your fine-tuned, specialized code
(years of research in numerics, HPC, RSE)

Where preCICE helps

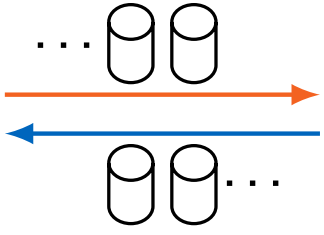


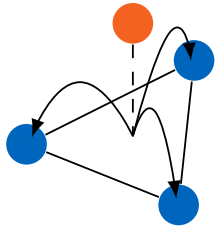
Communication

Options:
- MPI ports (fast)
- TCP sockets (robust)

Fully-parallel, peer-to-peer

Where preCICE helps





Communication

Options:
- MPI ports (fast)
- TCP sockets (robust)

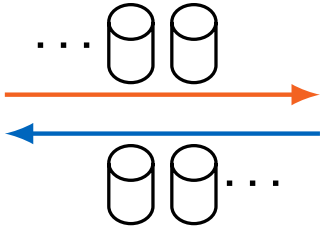
Fully-parallel, peer-to-peer

Data mapping

Options:
- radial-basis functions
- projection-based
- conservative/consistent
- direct mesh access

Compute on any side

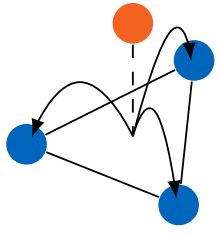
Where preCICE helps



Communication

Options:
- MPI ports (fast)
- TCP sockets (robust)

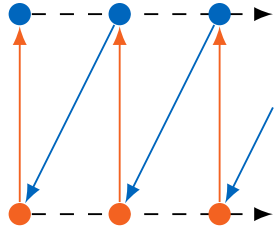
Fully-parallel, peer-to-peer



Data mapping

Options:
- radial-basis functions
- projection-based
- conservative/consistent
- direct mesh access

Compute on any side

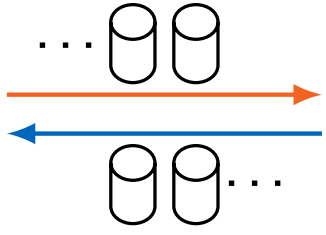


Coupling schemes

Options:
- serial / parallel
- explicit / implicit
- compositional, multi
- IQN, Aitken, ...

Same high-level API
Configurable at runtime

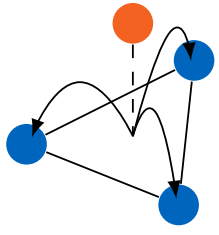
Where preCICE helps



Communication

Options:
- MPI ports (fast)
- TCP sockets (robust)

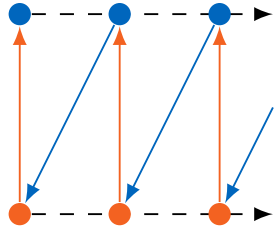
Fully-parallel, peer-to-peer



Data mapping

Options:
- radial-basis functions
- projection-based
- conservative/consistent
- direct mesh access

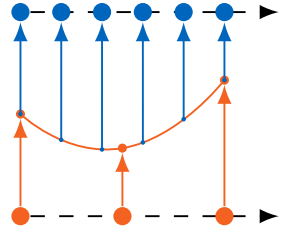
Compute on any side



Coupling schemes

Options:
- serial / parallel
- explicit / implicit
- compositional, multi
- IQN, Aitken, ...

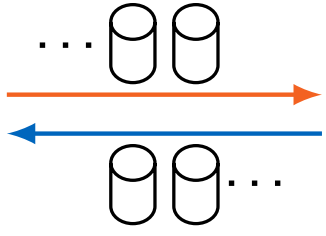
Same high-level API
Configurable at runtime



Time interpolation

Options:
- waveform iteration

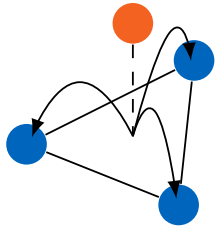
Where preCICE helps



Communication

Options:
- MPI ports (fast)
- TCP sockets (robust)

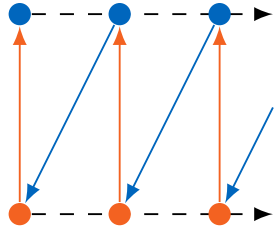
Fully-parallel, peer-to-peer



Data mapping

Options:
- radial-basis functions
- projection-based
- conservative/consistent
- direct mesh access

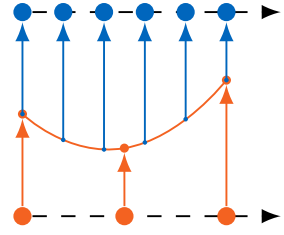
Compute on any side



Coupling schemes

Options:
- serial / parallel
- explicit / implicit
- compositional, multi
- IQN, Aitken, ...

Same high-level API
Configurable at runtime

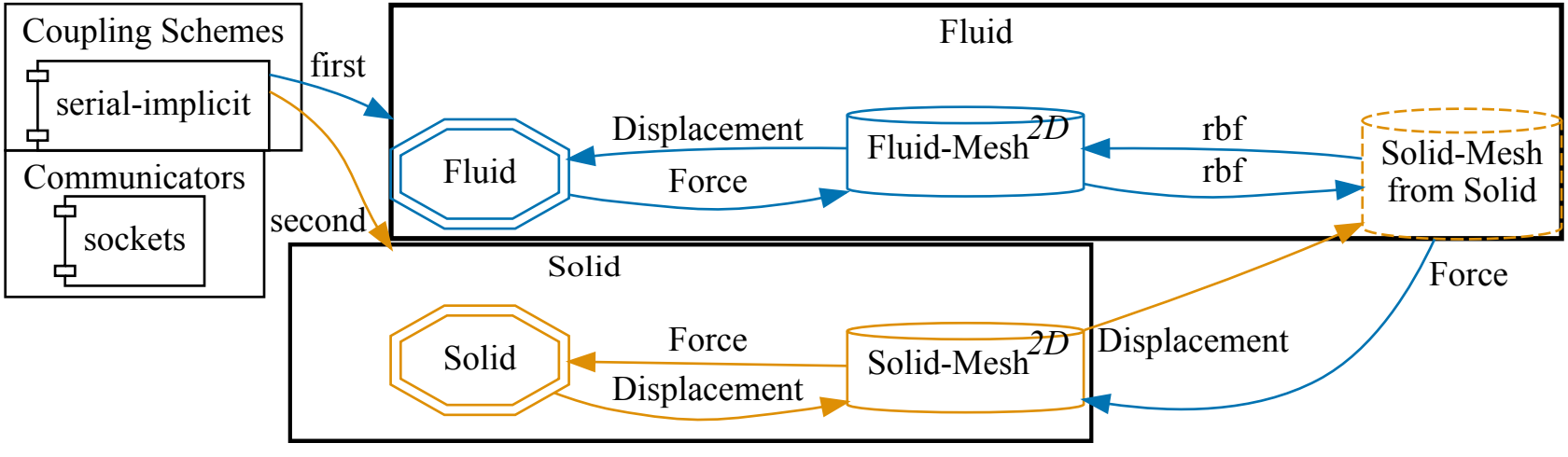


Time interpolation

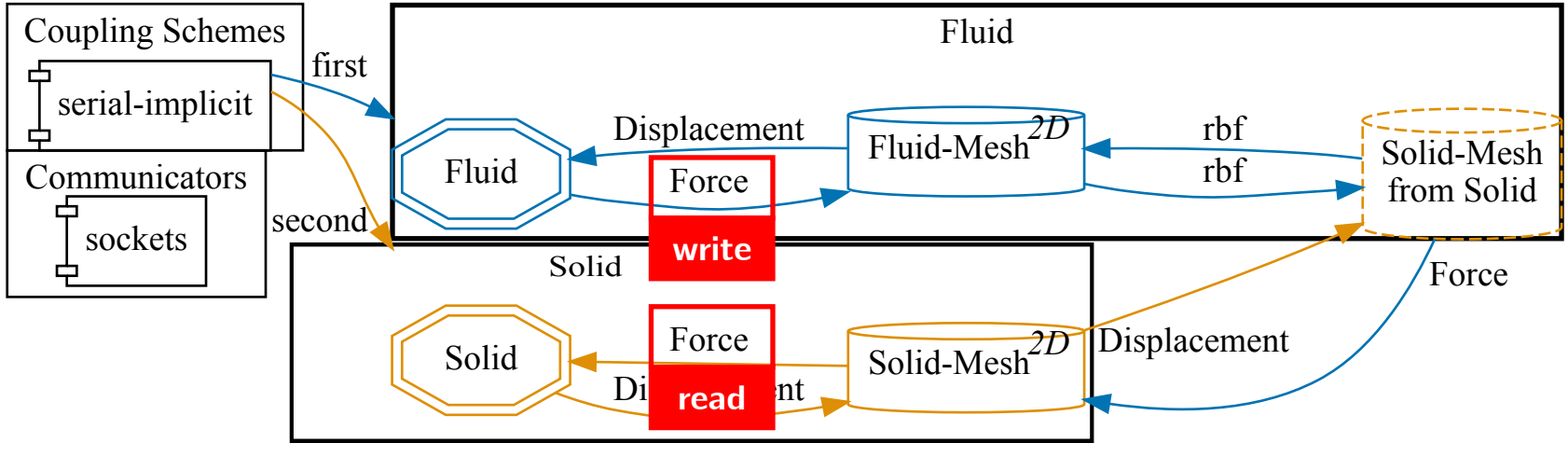
Options:
- waveform iteration

http://doi.org/
10/m9d8

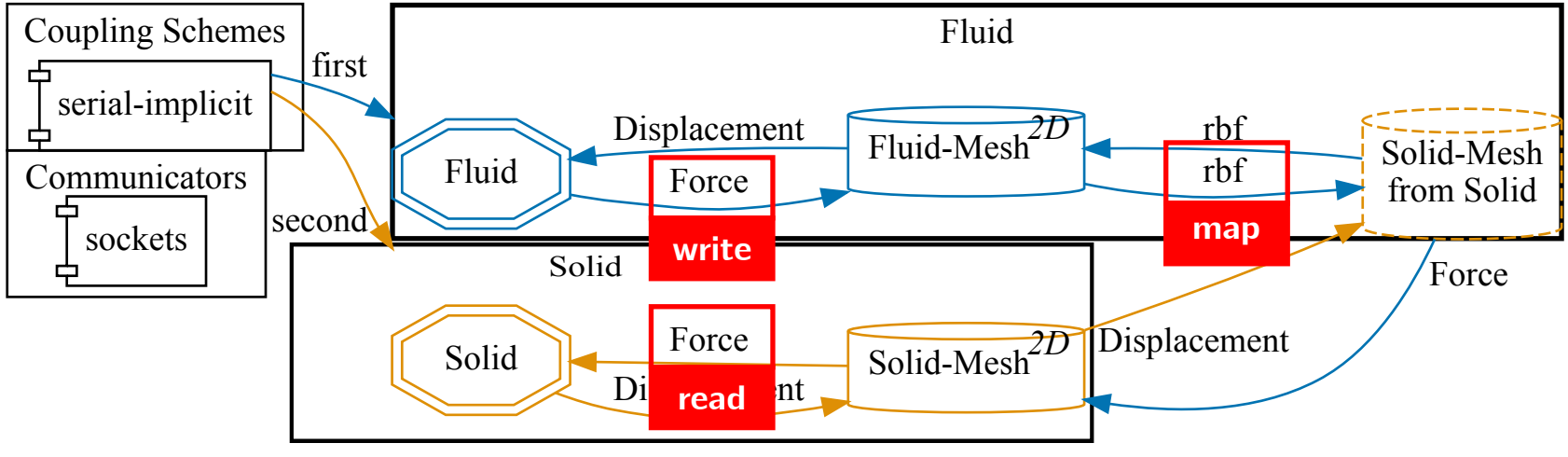
An FSI example setup



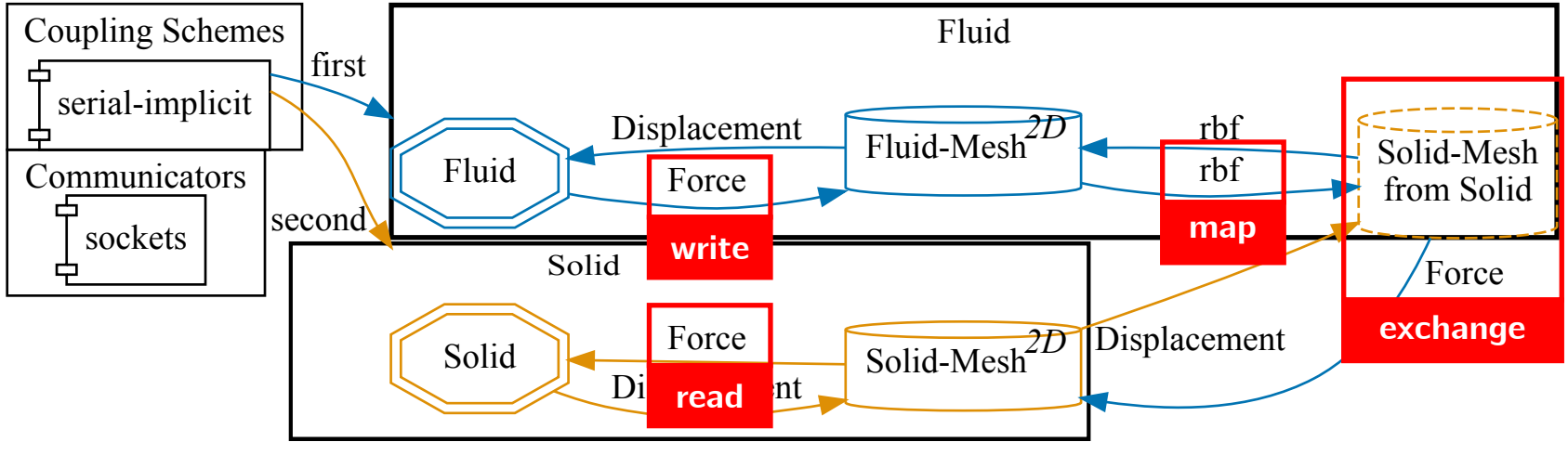
An FSI example setup



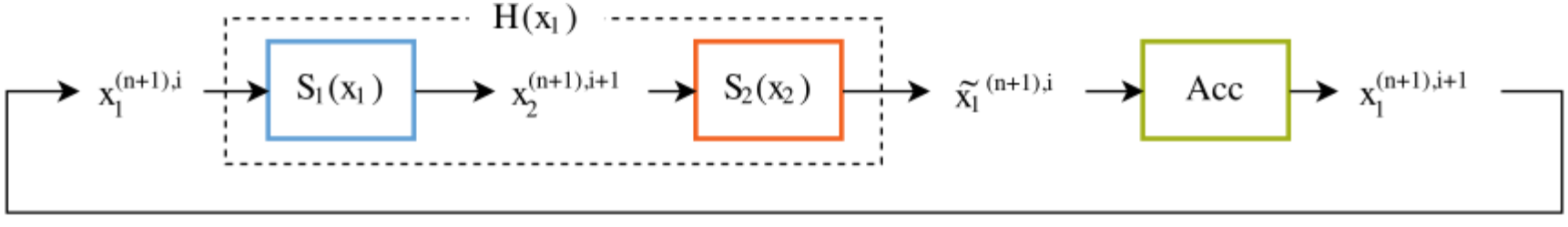
An FSI example setup



An FSI example setup



Implicit coupling algorithm (serial)



$$i \leftarrow i + 1, \quad S_1 \leftarrow S_1^{(n)}, \quad S_2 \leftarrow S_2^{(n)}$$

Acc: constant / Aitken under-relaxation, Anderson acceleration (IQN).
Also explicit, parallel-implicit, and multi-coupling schemes.

What people do with preCICE

From fluid-structure interaction to new directions

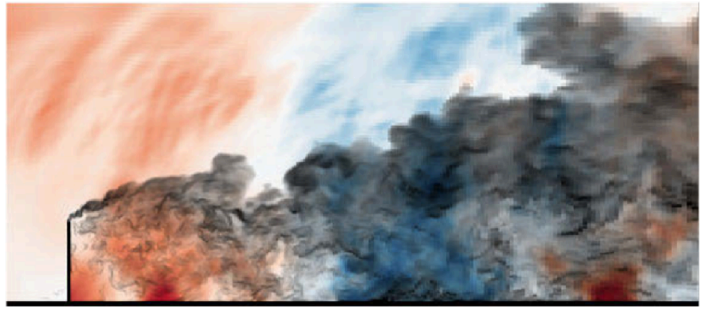
What people do with preCICE (1)

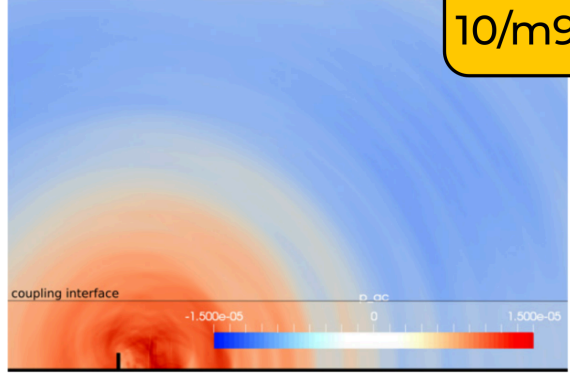
Home › Software for Exascale Computing - SPPEXA 2016-2019 › Conference paper

ExaFSA: Parallel Fluid-Structure-Acoustic Simulation

Florian Lindner, Amin Totounferoush, Miriam Mehl ✉, Benjamin Uekermann, Neda Ebrahimi Pour, Verena Krupp, Sabine Roller, Thorsten Reimann, Dörte C. Sternel, Ryusuke Egawa, Hiroyuki Takizawa & Frédéric Simonis

Conference paper | Open Access | First Online: 31 July 2020





http://doi.org/
10/m9d9

Kolb, Elena (2022):
Aeroacoustic Simulation of Turbulent Fluid-Structure Interactions at Low Mach Numbers.
In: Ingenieurwissenschaften, Berlin, Verlag Dr. Hut, TU Darmstadt, ISBN 978-3-8439-5061-9, [Dissertation]

Mariño Salguero, Jessica Marcela (2021):
Numerical simulation of free surface flows interacting with flexible structures. (Verlagsversion)
Darmstadt, Technische Universität, → DOI: 10.26083/tuprints-00019193, → Offizielle URL, [Dissertation]

ExaFSA SPPEXA program, TU Darmstadt (Germany)

What people do with preCICE (2)

Massively Multiscale Modeling using NASA Multiscale Analysis Tool through Partitioned Task-Parallel Approach

Ibrahim Kaleel, Trenton M. Ricks, Peter A. Gustafson, Evan J. Pineda, Brett A. Bednarcyk and Steven M. Arnold

AIAA 2023-2027
Session: Multiscale Modeling II

Published Online: 19 Jan 2023 • https://doi.org/10.2514/6.2023-2027

Read Now | Tools | Share

Abstract:

View Video Presentation: **https://doi.org/10.2514/6.2023-2027.vid**

NASA Multiscale Analysis Tool (NASMAT) is a "plug and play" software package that allows users to conduct massively multiscale modeling of hierarchical and nonlinear materials. This work extends the scalability and improves the High Performance Computing friendliness of NASMAT by adopting a partitioned task-parallel approach. The interoperability of NAS- MAT with external software is enhanced through integration with preCICE, an open source library for multiphysics coupling in a partitioned manner. Enhancement through preCICE al- lows for easy integration of NASMAT to other macro solvers and dissociates the parallelization strategy adopted within NASMAT from the macro solver. A task-parallel framework based on the master-worker approach is implemented as the parallelization scheme. The scheme accounts for the hierarchy of multiple scales (task-dependence) and the heterogeneous nature (dynamic load balancing) of computations. The applicability and scalability of the frame- work are evaluated by analyzing multiscale problems with varying degrees of complexity and computational loads.

Figures | References | Related | Details

SCITECH FORUM

AIAA SCITECH 2023 Forum

23-27 January 2023

National Harbor, MD & Online

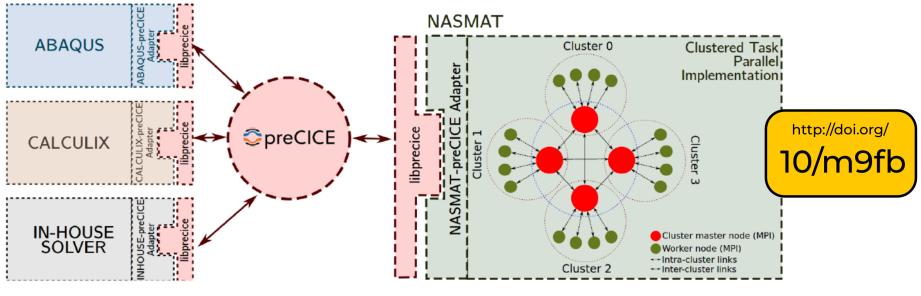
https://doi.org/10.2514/6.2023-2027

Crossmark

Information

This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

PDF



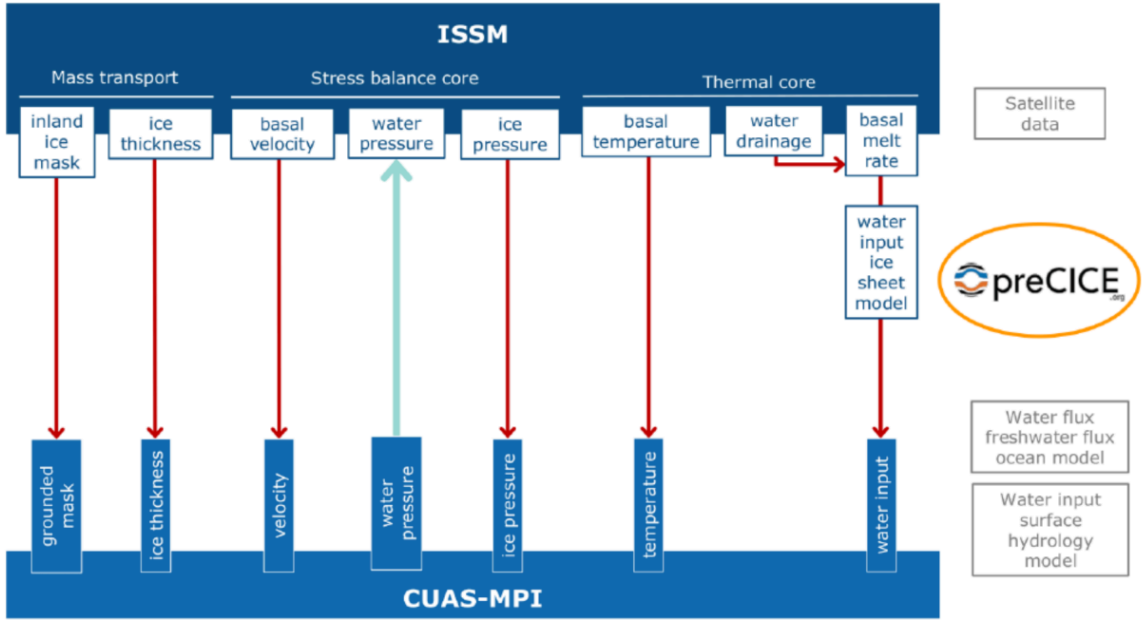
Hierarchical materials, NASA Glenn Research Center (USA)

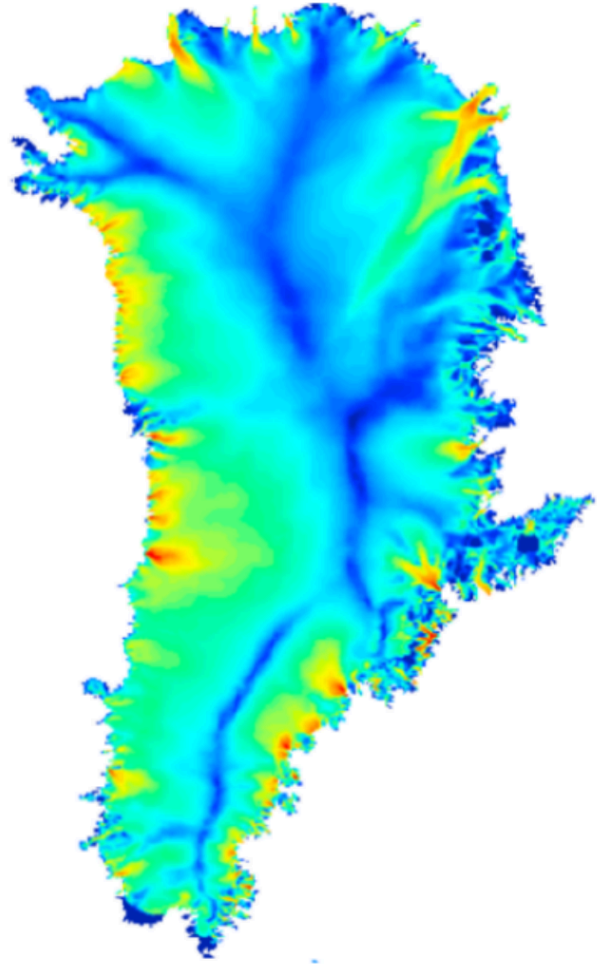
What people do with preCICE (3)

A preCICE-Interface for the Ice-sheet and Sea-level System Model (ISSM)

Yannic Fischler
yannic.fischler@tu-darmstadt.de
preCICE Workshop 2023

http://doi.org/10/m9fc





Ice-sheets and subglacial hydrology, AWI (Germany)

What people do with preCICE (4)

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

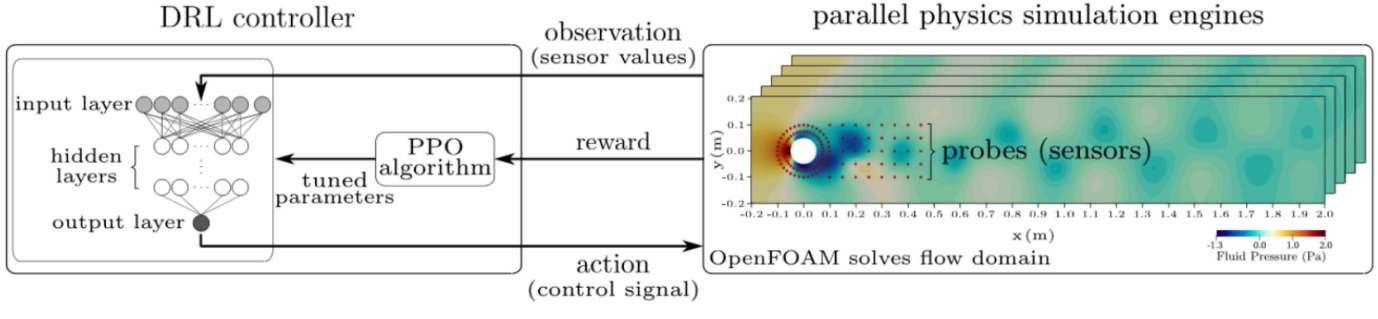
Original software publication

Gym-preCICE: Reinforcement learning environments for active flow control

Mosayeb Shams *, Ahmed H. Elsheikh

Heriot-Watt University, Edinburgh, United Kingdom

http://doi.org/
10/n3rd



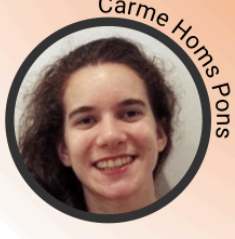
Active flow control with ML, Heriot-Watt University (UK)

Current research

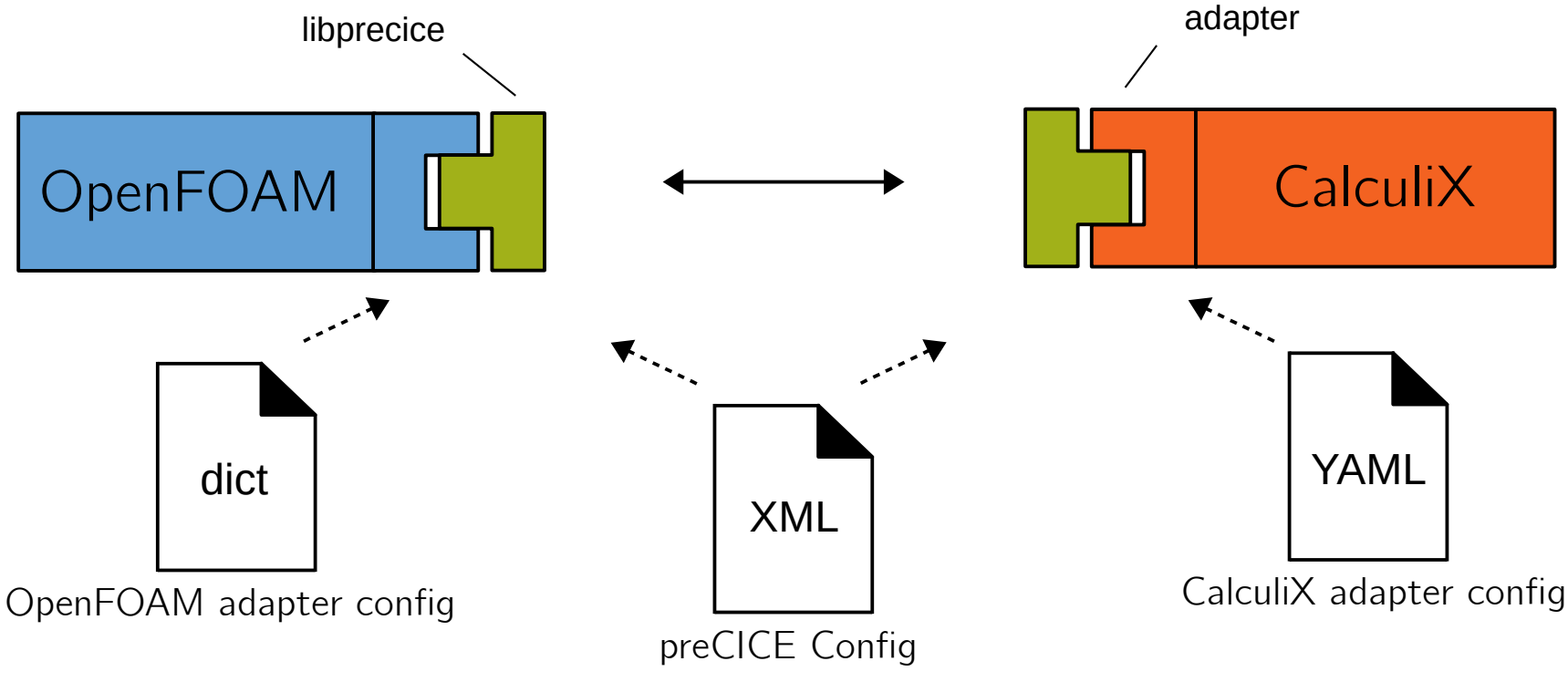
University of Stuttgart
Germany

Technical University of Munich

preCICE contributors + vibrant and rapidly growing community of



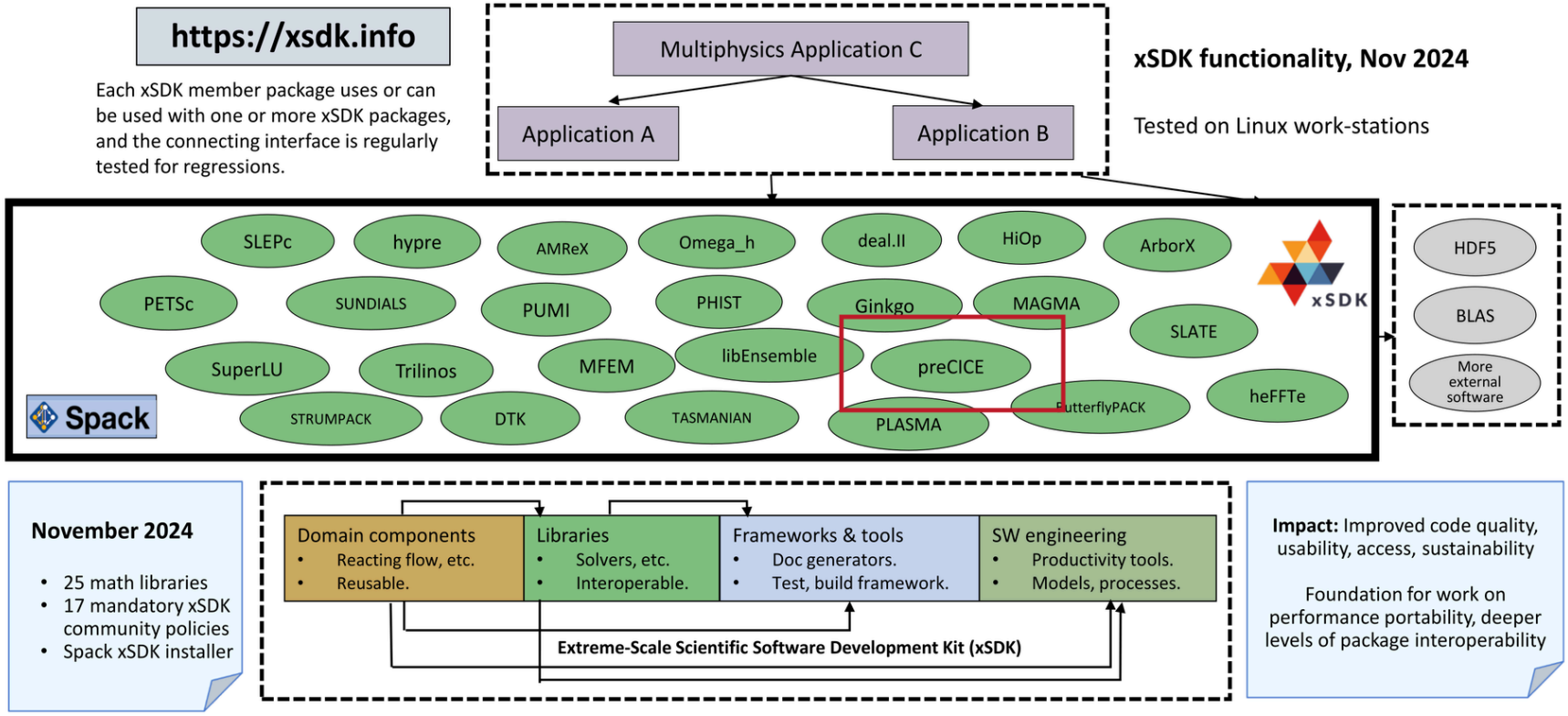
Defining community standards



Overview of an FSI setup: How easy is it to replace CalculiX by FEniCS?

xSDK Version 1.1.0: November 2024

https://xsdk.info



We can learn from the Extreme-Scale Development Toolkit: xsdk.info/packages/

Save the date: preCICE Workshop 2025, Hamburg, Sep 8-12

Community

Overview & news
preCICE workshops
preCICE minisymposia
Support preCICE
Training
Stories
Contributors
Contribute
Contribute to preCICE
Adapter guidelines
Application case guidelines
Community channels

Guidelines for adapters

Summary: Quality guidelines and standards for preCICE adapters and related tools

Table of Contents

Edit me

Updated 23 Jan 25

v0.2, published on December 9, 2024

Are you developing a preCICE adapter or related tool? Follow these guidelines to make your adapter easier to publish, easier to integrate with the rest of the preCICE ecosystem, and more useful for the community.

We differentiate between adapters and application cases: For example, we consider the "Nutils adapter" to be a collection of application cases instead of an adapter. An adapter can be a stand-alone software project, but can also be a class, a patch, or something else with significant scholarly effort compared to the uncoupled solver. Other tools that interact with the preCICE API or configuration files can also be considered here, on a case-by-case basis.

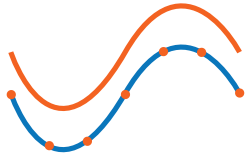
For the guidelines, we distinguish between **metadata** and **best practices**. The best practices are split between required and optional. Adapters fulfilling all the required best practices are listed on the preCICE website as adapters conforming to the preCICE standards. Additional criteria bring further benefits and are visible individually for each adapter. Adapters not fulfilling all the required criteria can still be listed here as legacy adapters. By submitting your adapter for review, you can expect a thorough check from the preCICE team on whether these guidelines are met.

precice.org/community-guidelines-adapters.html

Current research

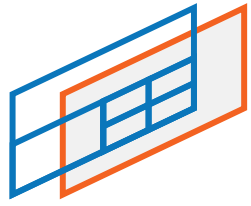
ecosystem

Gerasimos Chourdakis



time interpolation

Benjamin Rodenberg

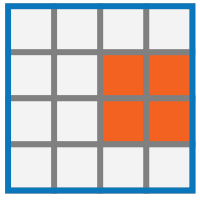


dynamic meshes

Frédéric Simonis

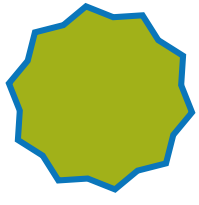
data mapping efficiency

David Schneider



adaptive multi-scale

Ishaan Desai



application: muscles

Carme Homs Pons

application: porous media

Jun Chen

💡 Time interpolation: See related talk by Niklas Kotarsky in MS205

Further research and development

- Coupling with Functional Mock-Up units
- Non-mesh-related global data exchange
- Volume coupling (overlapping domains)
- Coupling with particle codes
- Automatic acceleration configuration

preCICE Workshop #6 2025

HSU Hamburg | September

Germany | 8 - 12

~50 participants - Training - Talks - Posters - User support sessions - World Café

Abstract submission till May 31 - Registration till July 15 - See `precice.org`

Resources

Start here: `precice.org`

preCICE Quickstart Docs Tutorials Community Blog About Search ... Search by algolia

Save the date: preCICE Workshop 2025, Hamburg, Sep 8-12

Welcome to preCICE

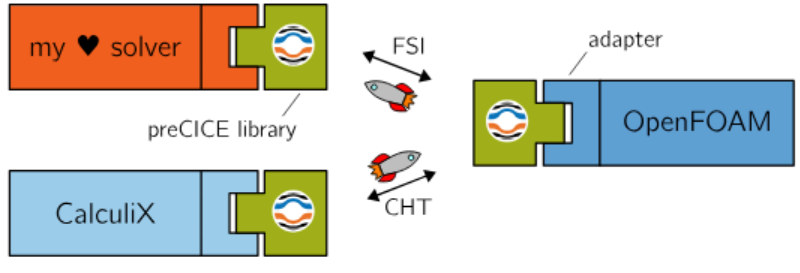
The coupling library for partitioned multi-physics simulations.

Star on Github 777 | Latest v3.1.2 (Jun 6, 2024) | Get started >

preCICE is an **open-source coupling library** for partitioned multi-physics simulations, including, but not restricted to fluid-structure interaction and conjugate heat transfer simulations.

Partitioned means that **preCICE couples existing programs/solvers** capable of simulating a subpart of the complete physics involved in a simulation. This allows for the high flexibility that is needed to keep a decent time-to-solution for complex multi-physics scenarios.

The software offers convenient methods for transient equation coupling, communication, and data mapping.



Everything is open and user-centered

- Developed in the open: github.com/precice
- Extensive documentation
- (Diamond) open-access publications
- Public recordings of talks

Extra effort to maintain and update

Further channels

precice.discourse.group

preCICECoupling

groups/9073912/

project/preCICE

fosstodon.org/@precicerecice

@precice.org

preCICE is free because of

Research Software
Sustainability

EXC 2075
SimTech

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 754462

and the code/issues/testing/documentation contributions of people like you (thank you!).

Summary

1. preCICE offers efficient algorithms for black-box coupling
2. New directions beyond FSI: looking for collaborations
3. Standardizing a community-driven bazaar of components

Gerasimos Chourdakis (US/TUM) + many more (see precice.org/about)

Extra slide: Example of using the API

```python
import precice, nutils

def main(young=4e6, density=3e3, poisson=0.3, nelems=2, solver_dt=0.(
    # Setup Nutils simulation... (mesh, equations, boundary condition

    # preCICE setup
    solver_process_index = 0
    solver_process_size = 1
    participant =
    precice.Participant("Solid", "../precice-config.xml",
                        solver_process_index, solver_process_size)
    vertex_ids =
    participant.set_mesh_vertices(mesh_name, wall.eval(ns.X))

    participant.initialize()

    while participant.is_coupling_ongoing():
```

Extra slide: Example of using the API

```python
import precice, nutils

def main(young=4e6, density=3e3, poisson=0.3, nelems=2, solver_dt=0.(
    # Setup Nutils simulation... (mesh, equations, boundary condition

    # preCICE setup
    solver_process_index = 0
    solver_process_size = 1
    participant =
    precice.Participant("Solid", "../precice-config.xml",
                        solver_process_index, solver_process_size)
    vertex_ids =
    participant.set_mesh_vertices(mesh_name, wall.eval(ns.X))

    participant.initialize()

    while participant.is_coupling_ongoing():
```

Extra slide: Example of using the API

```
    # preCICE setup
    solver_process_index = 0
    solver_process_size = 1
    participant =
    precice.Participant("Solid", "../precice-config.xml",
                        solver_process_index, solver_process_size)
    vertex_ids =
    participant.set_mesh_vertices(mesh_name, wall.eval(ns.X))

    participant.initialize()

    while participant.is_coupling_ongoing():
            if participant.requires_writing_checkpoint():
                checkpoint = timestep, arguments

            precice_dt = participant.get_max_time_step_size()
            dt = min(precice_dt, solver_dt)
```

Extra slide: Example of using the API

```
 9      participant =
10      precice.Participant("Solid", "../precice-config.xml",
11                          solver_process_index, solver_process_size)
12      vertex_ids =
13      participant.set_mesh_vertices(mesh_name, wall.eval(ns.X))
14
15      participant.initialize()
16
17      while participant.is_coupling_ongoing():
18              if participant.requires_writing_checkpoint():
19                  checkpoint = timestep, arguments
20
21              precice_dt = participant.get_max_time_step_size()
22              dt = min(precice_dt, solver_dt)
23
24              # read forces from participant at the end of the timestep
25              force = participant.read_data("Solid-Mesh", "Force", vert
```

Extra slide: Example of using the API

```
19                  checkpoint = timestep, arguments
20
21              precice_dt = participant.get_max_time_step_size()
22              dt = min(precice_dt, solver_dt)
23
24              # read forces from participant at the end of the timestep
25              force = participant.read_data("Solid-Mesh", "Force", ver
26
27              timestep += 1
28
29              arguments = dict(dt=dt, u0=arguments['u'], v0=arguments[
30              arguments = solver.solve_linear('u:testu,v:testv', res, a
31
32              # write displacements to participant
33              write_data = wall.eval(ns.u, **arguments)
34              participant.write_data("Solid-Mesh", "Displacement", ver
35
```

Extra slide: Example of using the API

```
            timestep += 1

            arguments = dict(dt=dt, u0=arguments['u'], v0=arguments[
            arguments = solver.solve_linear('u:testu,v:testv', res, a

            # write displacements to participant
            write_data = wall.eval(ns.u, **arguments)
            participant.write_data("Solid-Mesh", "Displacement", vert

            # do the coupling
            participant.advance(dt)

            # read checkpoint if required
            if participant.requires_reading_checkpoint():
                timestep, arguments = checkpoint

    participant.finalize()
```

Extra slide: Example of using the API

```
27              timestep += 1
28
29              arguments = dict(dt=dt, u0=arguments['u'], v0=arguments[
30              arguments = solver.solve_linear('u:testu,v:testv', res, a
31
32              # write displacements to participant
33              write_data = wall.eval(ns.u, **arguments)
34              participant.write_data("Solid-Mesh", "Displacement", vert
35
36              # do the coupling
37              participant.advance(dt)
38
39              # read checkpoint if required
40              if participant.requires_reading_checkpoint():
41                  timestep, arguments = checkpoint
42
43      participant.finalize()
```