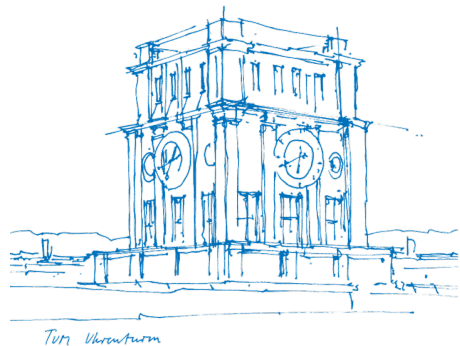
preCICE v2.0 and beyond

Technical University of Munich
Department of Informatics
Chair for Scientific Computing
Frédéric Simonis
17.02.2020



preCICE v2.0

Features

- Two-level Initialization
- Faster Initialization
- XML Reference
- Unified Configuration

Usability

- Simplified API and Configuration
- Clarified Naming
- Config Visualizer

Building and Packaging

- xSDK Member
- Repository Restructure
- External Bindings
- Improved Building

Two-level Initialization

- No mesh gather on master
- Allows for larger meshes
- Does not yet cover all corner-cases
- Enable using

```
<m2n use-two-level-init="on">
```

Today 18:00, Two-Level parallel initialization in preCICE, Amin Totounferoush

Further Improved Initialization Times

- Improved Nearest-Projection for common cases
- Fewer allocations
- Reduced memory footprint
- Locality speeds-up mesh operations
- Speed-up of computational part ≈ 2

XML Reference

- Now actually readable!
- Including examples
- Visit the Wiki: precice/wiki/XML-Reference
- Generate it yourself:

```
binprecice md > ref.md
```

mesh

Surface mesh consisting of vertices and (optional) of edges and triangles (only in 3D). The vertices of a mesh can carry data, configured by tag . The mesh coordinates have to be defined by a participant (see tag).

Example:

```
<mesh name="{string}" flip-normals="0">
  <use-data name="{string}"/>
</mesh>
```

Attribute	Type	Description	Default	Options
name	string	Unique name for the mesh.	*none*	none
flip-normals	boolean	Flips mesh normal vector directions.	`0`	none

Valid Subtags:

- use-data `0..*`

Unified Configuration

One configuration for parallel and serial:

- Implied `<master:mpi-single>` tag
- `<mapping:rbf>` selects PETSc-variant when needed
- `<m2n distribution-type="point-to-point">` no longer required

No need to switch configs!

Simplified API

```
SolverInterface interface(solverName, commRank, commSize);
interface.configure(configFileName);
// ...
interface.initialize(configFileName);
```

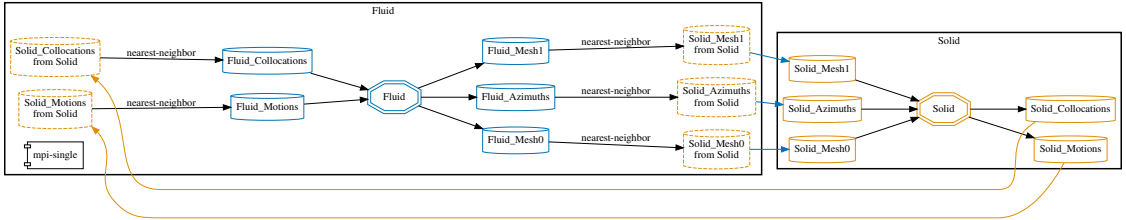
```
SolverInterface interface(solverName, configFileName, commRank, commSize);
// ...
interface.initialize(configFileName);
```

Clarified Naming

Renamed:

- `<post-processing>` to `<acceleration>`
- “timestep” to “time window” in interface and config
- “fulfilledAction” to “markActionFulfilled”
- `<use-mesh geometric-filter="...">` options to contain actor

Config Visualizer



precice/config-visualizer

xSDK Member Since 0.5.0

```
$ spack install precice
$ spack install xsdk

$ spack load precice
```

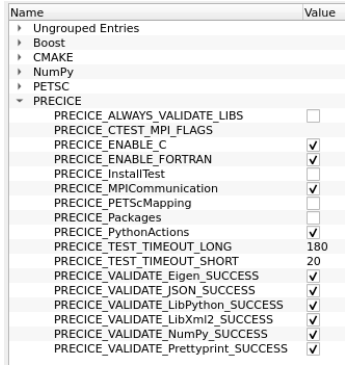
Improved Building and Testing

- Defaults to `Debug` build
- C++11 library check for Intel Compiler users
- Restructured and clarified output
- Grouped tests by component

Clarified CMake Variables

Prefixed and expressive

- `PRECICE_MPICommuniction`
- `PRECICE_PETScMapping`
- `PRECICE_PythonActions`
- `PRECICE_ENABLE_C`
- `PRECICE_ENABLE_FORTRAN`



Restructured Repository

The Pitchfork Layout (PFL)
Colby Pike vectorofbool@gmail.com

examples/ contain solverdummy *installed*
extras/ contain user tools
tools/ contain developer tools

```
$ pwd
/home/fsimonis/sd

$ cmake /usr/share/precice/examples/solverdummies/cpp
-- The CXX compiler identification is Clang 9.0.0
   ...
-- Found Threads: TRUE
-- Configuring done
-- Generating done
-- Build files have been written to: /home/fsimonis/sd

$ make
[ 50%] Building CXX object
↪  CMakeFiles/solverdummy.dir/solverdummy.cpp.o
[100%] Linking CXX executable solverdummy
[100%] Built target solverdummy
```

Restructured Bindings

C/Fortran	native, no change
Fortran 2003	precice/fortran-module
pyprecice	precice/python-bindings Official release! `pip install --user pyprecice`
Matlab	precice/matlab-bindings New bindings!

- Including documentation
- Including examples
- Separate releases

Porting Guide

Full guide can be found in the Wiki.

1. Building
2. API
3. Configuration

1. Building

1. Use CMake instead of SCons
2. Rename CMake variables

Old Variables	New Variables
`MPI`	`PRECICE_MPICommuniction`
`PETSC`	`PRECICE_PETScMapping`
`PYTHON`	`PRECICE_PythonActions`

2. API

1. Merge constructor and `configure()`

```
-SolverInterface interface(solverName, commRank, commSize);
-interface.configure(configFileName);
+SolverInterface interface(solverName, configFileName, commRank, commSize);
```

2. Rename `isTimestepComplete` to `isTimeWindowComplete`

```
-interface.isTimestepComplete()
+interface.isTimeWindowComplete()
```

3. Rename `fulfilledAction` to `markActionFulfilled`

```
-interface.fulfilledAction(name)
+interface.markActionFulfilled(name)
```

3. Configuration - Changes

1. Remove `<master:mpi-single>` tags
2. Remove `distribution-type` in `<m2n distribution-type="..." />`
3. Rename `<post-processing>` to `<acceleration>`

3. Configuration - Rename Options

Tag	Old Option	New Option
`cplscheme`	`timestep-length`	`time-window-size`
`cplscheme`	`max-timesteps`	`max-time-windows`
`acceleration`	`timesteps-reused`	`time-windows-reused`
`acceleration`	`reused-timesteps-at-restart`	`reused-time-windows-at-restart`
`export`	`timestep-interval`	`every-n-time-windows`

Tag	Attribute	Old Value	New Value
`use-mesh`	`geometric-filter`	`broadcast-filter`	`on-slaves`
`use-mesh`	`geometric-filter`	`filter-first`	`on-master`

Roadmap

Major Upcoming Features

- PhD topic
- One primary developer
- Long term development
- Collaboration welcome!

Minor Anticipated Features

- Generally not scheduled
- Occasional student topic
- Contributions welcome!

Extended Two-level Parallel Initialization

Developer: Amin Totounferoush

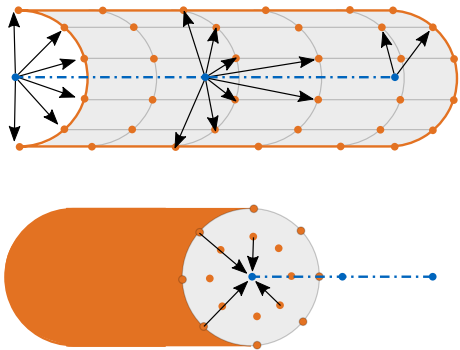
- Cover all corner cases
- Test extensively
- Enable by default

Today 18:00, Two-level parallel initialization in preCICE, Amin Totounferoush

Geometric Multi-Scale Mapping

Developer: Gerasimos Chourdakis

- Mapping between 1D-2D, 1D-3D
- Physically meaningful mapping
- Use-case e.g. CFD



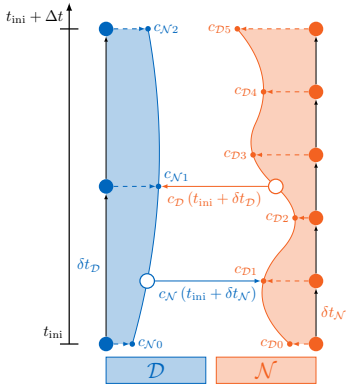
Chourdakis, G. (2019). Coupling OpenFOAM to different solvers, physics, models, and dimensions using preCICE. In 14th OpenFOAM Workshop.

Consistent Time Interpolation

Developer: Benjamin Rüth

- Provide data for any t
- Interpolation happens black-box
- More accurate and efficient implicit coupling
- Invisible to the user

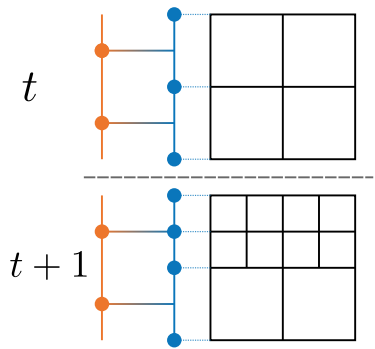
Tomorrow 10:00, High-order and multi-rate time stepping with preCICE, Benjamin Rüth



Dynamic Adaptive Meshes

Developer: Frédéric Simonis

- Reset a mesh on demand
- Allows redefining in every time-step
- Avoid complete reinitialization
- Adaptive meshes on interface
- Dynamic meshes



Minor Anticipated Features

Contributions Welcome!

github.com/precice/precice/contribute

Nearest Projection for Volume Coupling

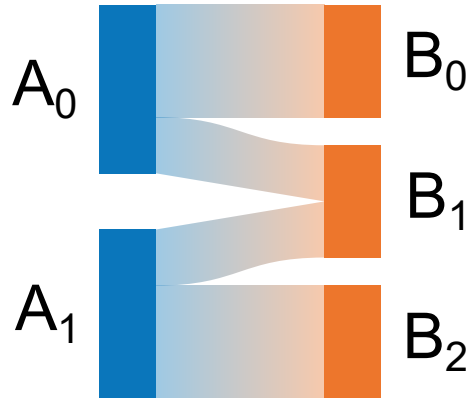
- Mapping on interface or volume
- Changes in mapping scheme
- 3D-Elements tetrahedra, octahedra

Support for Quads

- Hexahedral meshes common
- How to project onto a Quad?
- How to control behaviour?
- Internal triangulation?

Contiguous Mapping

- Matching meshes: Currently NN Mapping
  We can do better!
- Matching meshes with identical vertex order?
- Order as in time of registration
- Mapping of sub-ranges



Bulk Functions for Setting Meshes

- Edges and triangles
- API input sanitization
- API calls guarantee consistent state
- Bulk functions vastly more efficient

```
void SolverInterface::setMeshEdges(
     int meshID,
     int size,
     const int* vertexIDs,
     int* edgeIDs);

void SolverInterface::setMeshTriangles(
     int meshID,
     int size,
     const int* edgeIDs);
```

Watch-Integral

- Track integral values over a mesh

```
<watch-integral mesh="MyMesh1" name="MyWatchIntegral" />
```

- Total force on a geometry in an FSI simulation
- Flow rate for a fluid-fluid coupling

Questions?

Frédéric Simonis
Technical University of Munich
simonis@in.tum.de