Coupling OpenFOAM to different solvers, physics, models, and dimensions using preCICE

Gerasimos Chourdakis, Technical University of Munich
Benjamin Uekermann, Gertjan van Zwieten, Harald van Brummelen
Eindhoven University of Technology

OpenFOAM provides a rich arsenal of single-physics solvers, while other software projects also offer a wide range of solvers for structural dynamics or heat transfer. Moreover, packages for special applications, such as nuclear reactor safety, hemodynamics, or flood simulations need to integrate 3D flow phenomena into their workflow, which is often built around 1D or 2D models.
The coupling library preCICE for partitioned multi-physics simulations can bring together different solvers, in a minimally invasive way. It allows them to communicate via MPI ports or TCP/IP sockets, it maps the boundary values using advanced methods such as RBF, and it couples them with Interface Quasi-Newton algorithms that accelerate the convergence. Its API is being used in a variety of well-known or in-house solvers, while official, user-ready adapters are provided for open-source packages such as OpenFOAM. The official OpenFOAM adapter supports conjugate heat transfer and fluid-structure interaction out-of-the-box, while it was recently extended to also support fluid-fluid coupling.
In this paper, we do not focus on general aspects of preCICE or the OpenFOAM adapter, but we pick a certain aspect of fluid-fluid coupling, for which we discuss recent activities: coupling 3D OpenFOAM to any external 1D fluid solver. For general information on preCICE and the OpenFOAM adapter, we refer to [1] and [3, 4], respectively.

1 Introduction

Surface coupling between 3D and 1D fluid solvers is a crucial ingredient for many applications, as it allows to resolve local 3D phenomena where necessary, while still managing large simulation domains that are mainly dominated by 1D phenomena. In hemodynamics, for example, singular 3D vessels where local details are of interest are coupled to the complete 1D circulatory network [8]. In the automotive sector, internal 3D combustion engines are coupled to 1D intake and exhaust systems [7]. In nuclear safety analysis, 3D fluid phenomena are coupled to 1D system codes [6]. As a last example, in urban flooding simulation, shallow water codes are coupled to pipe networks of the sewer system [10]. Often, the complexity of such a 3D-1D coupling is underestimated, but the open questions and challenges are numerous and include, non-exclusively:

- Where is the ideal location of the 3D-1D interface? Naturally, as close as possible to the physical 3D phenomena to save computational costs and to simplify the meshing. However, putting it too close might violate the modeling assumptions of the 1D solver (e.g. how should we deal with recirculation?).
- What are valid solver coupling conditions?
- How to treat two-phase/slug flow?
- How to design a flexible software system and numerical framework reusing existing codes?

There are already specialized 3D-1D coupling approaches implemented with OpenFOAM. The general grid interface allows to internally couple 3D and 1D OpenFOAM meshes [11]. 1D solvers can directly be implemented as an OpenFOAM boundary condition [11]. The 1D solver can be called as a library from a modified OpenFOAM solver [6]. For the in-house 3D-1D coupling from Polytecnico di Milano [7], we could not find architectural details.

The aim of our research is to develop a general 3D-1D interface by using and extending the coupling library preCICE [1]. By "general", we mean that the coupling should be independent of the coupled codes and the application. In particular, the 1D solver should be a stand-alone code. We want to treat the coupled codes as black boxes. For 3D-1D coupling, this means that the 3D solver does not know whether it is coupled to another 3D or to a 1D partner. To this end, we need to separate the coupling physics from the geometric mapping.

Black-Box Coupling Numerically, black-box coupling is the extreme case of a partitioned (or staggered) coupling: only minimal information and no discretization details from the solvers should be used, to allow for maximal flexibility. We, therefore, abstract the solvers as input-output operators, e.g. the first fluid solver in a coupled setup as $F_1 : U \mapsto P$, where U is the discretized vector of all velocity values on the coupling surface set as Dirichlet boundary condition in F_1, and P is the discretized vector of all pressure values at coupling interface after one time step. We neglect the time dependence $F_1 \equiv F_1^t$ with this notation. Let the second fluid solver $F_2 : P \mapsto U$ complement the first one by taking pressure values as Neumann boundary condition and returning velocity values after one time step. We assume matching meshes at the coupling interface. If not, we can apply a mapping operator [5], which we neglect here for simplicity. We also neglect gradient values at the coupling interface, which the operators could also include. To advance in time, F_1 and F_2 can either be applied after another, in a Gauss-Seidel kind of way, or simultaneously, in a Jacobi kind of way. We only consider Jacobi coupling in this paper. Last, we distinguish between explicit and implicit coupling. Explicit coupling solves both subdomains only once per timestep. Implicit coupling sub-iterates between both until convergence of the interface residual. Please note that the term implicit coupling is sometimes also used for monolithic coupling approaches [11]. For implicit coupling, we still apply a partitioned black-box coupling. For Jacobi coupling, we aim to solve the fixed-point equation

$$(F_1(U), F_2(P)) = (P, U)$$

in every timestep. We solve the residual of this interface system by Newton's method. As no discretization details of the solvers are available (e.g. no shape functions for finite element solvers), we approximate the Jacobian by means of previous iterates, possibly also from previous timesteps, and, thus, end up with a quasi-Newton method. For more details of the black-box coupling and the quasi-Newton methods, see Bungartz et al. [2].

In the remainder of the paper, we first classify 3D-1D coupling approaches by their geometric mapping in Section 2. Section 3 details the necessary software extension in preCICE and OpenFOAM. Section 4 validates the implementation, simulating water hammer in a hydraulic pipe flow. Finally, we conclude the paper and list future work in Section 5.

2 Classification of 3D-1D Coupling

For a 3D-1D coupling, we need to create a mapping between the shared boundaries of the 3D and the 1D domain: a surface and one or multiple points. In this paper, we only assume coupling between two perfectly matching cylindrical domains with constant diameter, as this is the simplest geometry that a 1D domain can represent. However, our proposed classification can also be extended to other setups.

We distinguish three different characteristics of a 3D-1D mapping: the geometric relation of the domains, the mapping direction, and the conservation of integral values. An example is shown in Figure 1.

Geometric Relation In the example of the cylinders, we can choose to couple their discs, as if we connected two pipes. We name this "axial mapping", as we map along the axis. Alternatively, we can choose to couple their curved surfaces, as if two pipes were nested. We name this "radial mapping", as we map along the radius. The difference is: do we map to one boundary point of the 1D domain or do we map to multiple internal points? We consider pipe-pipe coupling as an example of axial mapping; we consider FSI with a 1D structure immersed in a 3D fluid as an example of radial mapping.

Direction In every possible scenario, we can choose to map from the 3D solver to the 1D solver, or the reverse. We name the 3D $\rightarrow$ 1D a "collect mapping" (as we collect information from multiple points to one); we name the 1D $\rightarrow$ 3D a "spread mapping" (as we spread information from one point to multiple), see Figure 1. The difference between the two is: do we need to reduce (3D $\rightarrow$ 1D) or to generate (1D $\rightarrow$ 3D, defective boundary condition [8]) information? Both situations are encountered in a bi-directional coupling of the above scenarios (e.g. 3D velocity/force vs 1D velocity/force) and there are multiple decisions to be taken when generating information (e.g. velocity profile). Wang et al. [11] also identify this as a characteristic, for which they propose the terms "reduce" and "expand".

Conservation of Integrals Similarly to coupling solvers of the same dimensions [5], we distinguish between mapping of non-integral (e.g. velocity) and mapping of integral values (e.g. force). We follow the name "consistent mapping" when information is replicated identically; we follow the name "conservative mapping" when information is mapped so that integrals are conserved.

The aforementioned characteristics lead to eight different combinations:

$\{\text{axial}, \text{radial}\} \times \{\text{collect}, \text{spread}\} \times \{\text{consis.}, \text{conserv.}\}$

3 Software

In this work we used OpenFOAM[1], preCICE[2], and the OpenFOAM adapter for preCICE[3].

preCICE is a free/open-source library for partitioned multi-physics coupling. It is minimally invasive, considers the solver a (numerical) black-box and official “adapters” for solvers allow to create plug-and-play multiphysics simulations from single-physics solvers [1]. While the coupling, communication, and (3D-1D) mapping are handled by the preCICE library, the different physics (boundary conditions) are handled by each adapter.

The official OpenFOAM adapter for preCICE [4] is an OpenFOAM function object. This way, the adapter can be loaded at runtime, without any code changes [3]. As shown in Figure 2, the physics-related part of the adapter is encapsulated into “modules” which convert the physical values from/to OpenFOAM to/from preCICE buffers. These modules are highly separated from the steering part of the adapter, which makes the adapter easy to extend to other types of coupled problems. The adapter already supports conjugate heat transfer and fluid-structure interaction, while a prototype for fluid-fluid coupling is presented here.

To realize 3D-1D fluid-fluid coupling, we need to modify both preCICE and the adapter. In the next subsection, we start by explaining the changes in the OpenFOAM adapter, followed by a description of the necessary extentions in preCICE [4].

3.1 Fluid-Fluid Module for the preCICE OpenFOAM Adapter

Every “module” of the OpenFOAM adapter provides methods to extract values from OpenFOAM patches and write them to a preCICE buffer, as well as methods to read values from a preCICE buffer and set the respective values or gradients of selected OpenFOAM patches. The fluid-fluid module (see Figure 2) adds support to read and write velocity, pressure, velocity gradient, and pressure gradient. It is currently tested with the incompressible solver `pimpleFoam` and with the compressible solver `sonicLiquidFoam`, while it is known to work with OpenFOAM 4.0 – 7 (openfoam.org) and v1706 – v1906 (openfoam.com).

It is important to understand that the reading is not implemented as OpenFOAM boundary conditions, retaining a minimally-invasive design. Instead, at every time step, the adapter forcefully

[1]The OpenFOAM Foundation, version 5.x, 20171030
[2]github.com/precice/precice, commit f12e5bd
[3]github.com/precice/openfoam-adapter, commit 857f18c
[4]These prototype implementations are not yet shipped with the normal releases of preCICE / OpenFOAM adapter.

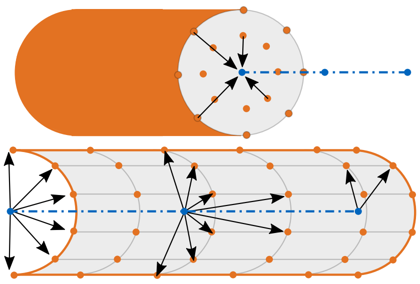


Figure 1: 3D-1D mapping types: axial-collect (top) and radial-spread (bottom) mapping.

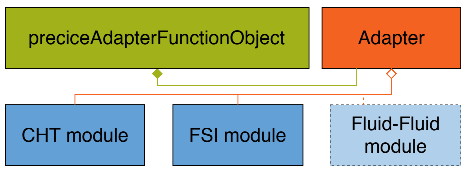


Figure 2: Overview of the OpenFOAM adapter design: a function object, with separate steering and (extendable) physics parts.

overwrites the boundary data. For this, it is important that the user sets a `fixedValue` boundary condition for patches that read velocity/pressure, or `fixedGradient` for patches that read gradients.

In the following, we give some implementation details. Reading pressure values on the face center of cell `i` of the interface patch with ID `patchID`:

```
p->boundaryFieldRef()[patchID][i]
=
buffer[bufferIndex++];
```

Writing pressure values is implemented similarly. Reading pressure gradients (normal derivatives):

```
scalarField & gradientPatch
=
refCast<fixedGradientFvPatchScalarField>
(
    p->boundaryFieldRef()[patchID]
).gradient();

forAll(gradientPatch, i)
{
    gradientPatch[i]
    =
    buffer[bufferIndex++];
}
```

Writing pressure gradients (normal derivatives):

```
scalarField gradientPatch
=
refCast<fixedValueFvPatchScalarField>
(
    p->boundaryFieldRef()[patchID]
).snGrad();

forAll(gradientPatch, i)
{
    buffer[bufferIndex++]
    =
    -gradientPatch[i];
}
```

Notice here the different target type in the `refCast`, the different method invoked to compute the gradient, as well as the sign-reverse operation.

Before continuing to more complex scenarios, we need to examine the validity of this approach. We simulate a simple partitioned pipe with `pimpleFoam`, as depicted in Figure 3. The pipe points to the z-direction, has a diameter of $10\,\mathrm{m}$, length of $40\,\mathrm{m}$, and is cut into left and right parts at $z = 20\,\mathrm{m}$. The kinematic viscosity and density of the fluid are $\nu = 10\,\mathrm{m}^2/\mathrm{s}$ and $\rho = 1\,\mathrm{kg}/\mathrm{m}^3$, respectively.

For the left pipe, we set fixed (zero) pressure gradient and fixed velocity value ($0.1\,\mathrm{m}/\mathrm{s}$) on the inlet and fixed (zero) pressure value and fixed (zero) velocity gradient on the outlet. For the right pipe, we use a similar configuration (inlet: $\partial_n p = 0$, $U = 0$, outlet: $p = 0$, $\partial_n U = 0$). In respect to coupling, the left pipe writes velocity (and in some cases pressure gradient) and reads pressure, with the right pipe complementing the left. In every case, we use implicit coupling, meaning we subiterate until convergence, as described in Section 1.

As shown in Figure 3 (left), the overall results for velocity look very good. Zooming in, however, we notice small oscillations and a velocity jump around the interface. Coupling only velocity and pressure leads to oscillations, but no velocity jump. When exchanging also the pressure gradient, the oscillations disappear, but an artificial source of velocity is introduced, clearly violating the mass balance. A similar pressure jump is also observed (not shown here). This offset seems to be reduced with a smaller time step size, as well as with a finer mesh, but it does not vanish. Even though the overall results look good, this issue needs to be addressed in future work. We suspect a technical issue with the gradient computations. Suggestions from the community are very welcome.

3.2 3D-1D Data Mapping in preCICE

From the mapping types described in Section 2, we present here two examples: *axial-collect-consistent* and *axial-spread-consistent*. This combination suffices for axial pipe-pipe coupling, in which we map pressure and velocity in a consistent manner.

The current prototype implementation comes with a few limitations: it only supports serial participants and it cannot yet take advantage of mesh connectivity (a feature used in other mappings). This means that, for the *axial-spread-consistent*, the boundary value of the 1D domain is copied to every vertex of the 3D domain:

$$v_i^{3D} = v^{1D} \quad \forall i = 1 \ldots n \ ,$$

where v_i^{3D} are the values on the 3D surface mesh, n the number of vertices on the 3D surface mesh, v^{1D} the one value on the 1D mesh. For the *axial-collect-consistent*, the values on the 3D vertices are averaged and assigned to the 1D boundary:

$$v^{1D} = \frac{1}{n}\sum_{i=1}^{n} v_i^{3D} \ .$$

If we also considered mesh connectivity, this averaging could be weighted by the surface area surrounding each vertex.

Figure 4 shows an excerpt of the preCICE configuration file, as adapted for the new mappings. Two participants are defined, but only the `3D-Solver` uses both interface meshes and computes both mappings. Both mappings use the same XML tag and they define their type and constraint as we already saw. The `from`, `to`, and `direction` follow the same principles as for other mapping types[5].

This is only our first approach to implementing 3D-1D mapping in preCICE. In the future, we want to implement all the combinations described in Section 2, while keeping the implementation generic and extending to parallel solvers. We also plan to apply these methods on two-phase flow coupling, a case for which we expect additional challenges.

4 Numerical Example: Water Hammer in Hydraulic Pipe Flow

We validate our prototype implementations by simulating water hammer in a hydraulic pipe flow. We consider the reservoir-duct-valve system from Wang et al. [11] and try to reproduce their results. This scenario has a pure 1D nature and a 1D simulation is completely sufficient to capture the physics. Still, this fact makes it also a good validation scenario for our 3D-1D coupling. The complete setups of all tests of this section are available online [6].

4.1 Equations and Setup

We assume laminar flow of a barotropic compressible fluid. The continuity and momentum equations read:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \ ,$$

$$\frac{\partial (\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) - \nabla \cdot \mu \nabla \mathbf{u} = -\nabla p \ ,$$

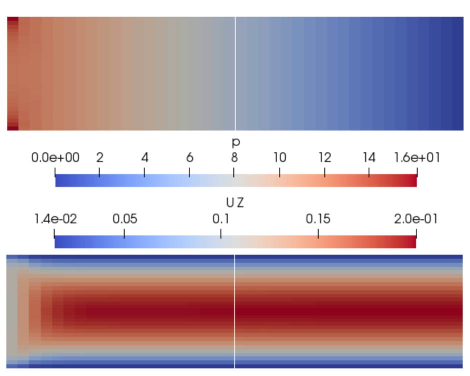
where ρ is the density, $\mathbf{u}$ the velocity, $\mu = 10^{-3}\mathrm{kg}/\mathrm{ms}$ the dynamic viscosity, and p the static pressure. A linearized barotropic relation closes the equations:

$$\rho \approx \rho_0 + \frac{1}{a^2}(p - p_0) \ ,$$

where $a = 1000\,\mathrm{m}/\mathrm{s}$ is the speed of sound, $\rho_0 = 10^3\,\mathrm{kg}/\mathrm{m}^3$ the reference density, and $p_0 = 101325\,\mathrm{Pa}$ the reference pressure.

[5] See preCICE wiki github.com/precice/precice/wiki/Mapping-Configuration.

[6] You may find the files on `http://go.tum.de/293358`.



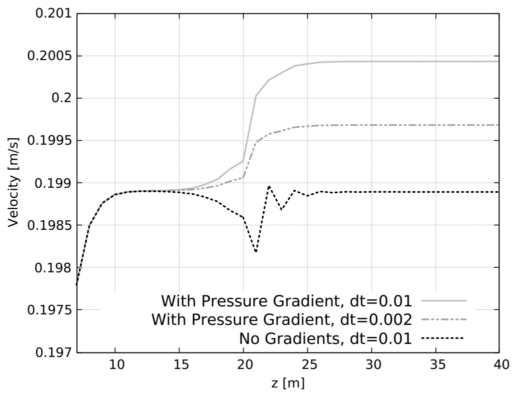


Figure 3: Pressure and z-Velocity on a slice of partitioned pipe (left), detail of the velocity magnitude along the centerline of the pipe (right).

```
<participant name="1D-Solver">
  <use-mesh name="1D-Mesh" provide="yes"/>
  ...
</participant>

<participant name="3D-Solver">
  <use-mesh name="3D-Mesh" provide="yes"/>
  <use-mesh name="1D-Mesh" from="1D-Solver"/>
  <write-data name="Velocity" mesh="3D-Mesh"/>
  <read-data name="Pressure" mesh="3D-Mesh"/>
  <mapping:axial-geometric-multiscale
     type="spread" from="1D-Mesh" to="3D-Mesh"
     direction="read" constraint="consistent"/>
  <mapping:axial-geometric-multiscale
     type="collect" from="3D-Mesh" to="1D-Mesh"
     direction="write" constraint="consistent"/>
</participant>
```

Figure 4: Excerpt of the preCICE configuration showing the geometric multi-scale mapping. Both mapping schemes are computed on the resources of the 3D solver. Only the 1D mesh is communicated.

The solver `sonicLiquidFoam` implements these equations and we received the setup from the authors of [11]. The domain is discretized with hexahedral cells and a finite volume scheme is used to solve the equations. This solver uses the PISO algorithm, solving for velocity and pressure in separate systems of equations. For time discretization, a 0.45 blend between backward Euler (corresponding to a coefficient of 1) and Crank-Nicolson (corresponding to a coefficient of 0) is applied as compromise between stability and minimal damping.

As 1D solver, we implement the exact similar set of equations in the finite element framework Nutils [9]. We use Taylor-Hood P2-P1 elements in 1D and solve for the velocity and pressure in a fully-coupled way. The time discretization is a crucial ingredient and is studied in the next section.

The reservoir-duct-valve system is depicted in Figure 5. We assume an infinite surface area of the reservoir and model it directly as a constant inlet boundary condition for the duct $p_{in} = 98100\,\mathrm{Pa}$. The opening valve is modeled through an outlet velocity which increases linearly from 0 to 1 m/s over 5 s and remains constant afterwards. The opening valve leads to pressure waves traveling through the duct, the so-called water hammer, which in practice can lead to damages on the duct system and, therefore, is of high interest. The duct is 1000 m long and has a constant square cross-section area of $1\,\mathrm{m}^2$. For the 3D solver, we model the outer duct wall with slip boundary conditions, neglecting friction resistance. The mesh cells have a constant size of 5 m along the duct for both the 3D and the 1D model. For the 3D model, we use a mesh size of 0.1 m perpendicular to the duct, resulting in 100 cells per cross-section. We use a time step size of 0.005 s and simulate the system for 20 s.

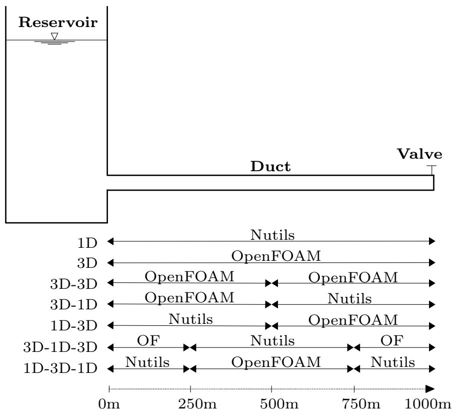


Figure 5: A reservoir-duct-valve system. A similar figure is already used by Wang et al. [11]

Figure 5 lists the different coupling combinations we test. For the 3D-1D mapping, we use the two axial variants that are introduced in Section 3.2: consistent-collect and consistent-spread. Please note that these mapping schemes do not strictly require a circular cross-section as depicted in Figure 1, but also work for the squared cross-section of the duct.

4.2 Influence of Time Integration

We validate our 1D implementation by comparing it to the 3D model, and, thereby, with the results of [11]. We observe a critical influence of the time discretization. We use a theta method and compare $\theta = 0.5$ (Crank-Nicolson), $\theta = 0.55$, and $\theta = 1.0$ (backward Euler). Figure 6 depicts the outlet pressure over time. In principle, all 1D results agree very well with the 3D results. Zooming in, we see that Crank Nicolson leads to small oscillations in the pressure, while backward Euler induces too much damping. $\theta = 0.55$ seems to be a good compromise. In general, the 3D results show a bit more damping and a lower amplitude. When using a smaller time step size for both solvers, the shape of the curves align more and more, while the gap in amplitude remains. Also refining in space has no influence on this gap, such that we conclude a small modeling disparity between the 3D and the 1D model.

4.3 Influence of Coupling Conditions

Before looking into the 3D-1D results, we want to carefully study our black-box coupling methodology and the influence of various choices in the coupling conditions and numerics. Therefore, we consider a 3D-3D coupling with two OpenFOAM instances and we compare it to a single, monolithic, OpenFOAM simulation; cf. Figure 5.

Explicit and Implicit Coupling We start by exchanging pressure and velocity values, but no gradients: velocity values from the right OpenFOAM solver to the left and pressure value from left to right. The inlet velocity gradient of the right OpenFOAM is implicitly set to zero, similar to the outlet pressure gradient of the left. Figure 7 compares explicit with implicit coupling, recalling that explicit coupling refers to a single exchange of data per time step, whereas implicit coupling refers to a repetition of every time step until convergence. As convergence measure, the relative interface residual needs to drop below 10^{-3}. For these settings, the implicit coupling only requires 1.97 iterations in average to converge. We observe that explicit coupling leads to a severe damping effect while implicit coupling matches the monolithic run very well.

Exchanging Gradients Next, we also exchange gradients: the pressure gradient from right to left and the velocity gradient from left to right. The implicit coupling becomes substantially harder to solve. 5.85 iterations are necessary in average, while often the maximum of allowed iterations, 30, is reached. Figure 8 shows that the additional gradient information leads to a slight overshoot of the outlet pressure, which increases over time. We get almost similar results if we neglect the velocity gradient, but still exchange the pressure gradient – the variant used by Wang et al. [11]. In principle, we get a similar behavior as already observed for incompressible flow in Section 3.1: gradient information has a negative influence on the coupling. It is to be noted that, in fact, the imposition of a pressure gradient in a fully-coupled formulation (such as Nutils) is non-standard. Further study in this direction is certainly needed. In contrast to the incompressible flow coupling, however, we cannot observe any negative influence when exchanging no gradient information, but only velocity and pressure values. Thus, for all further tests of this manuscript, we apply these simpler coupling conditions.

Quasi-Newton Asymmetry Last, we analyze the influence of the 3D-1D asymmetry on the quasi-Newton coupling. Should we compute the quasi-Newton acceleration on the 3D coupling mesh or on the single 1D coupling vertex? For the latter, the size of the fixed-point operator as introduced in Section 1 is four: three velocity components and one pressure component at the coupling vertex. Two velocity components are almost zero, giving us essentially a system with dimension two. For the 3D coupling mesh, we get a size of four times the number of 3D vertices at the coupling boundary. The values at each of these vertices are, however, nearly the same (as we deal with pure 1D physics), reducing the dimension of the complete problem again to four and then two. To analyze which choice is favorable, we simulate the 1D-3D configuration (cf. Figure 5) with both variants. As expected, the quasi-Newton coupling is able to cope with both situations. We get the same results for both variants and almost the same number of iteration – slightly under two in average. To compute the acceleration on the 1D mesh is, therefore, the superior choice as it is cheaper to compute.

4.4 Comparison of Different 3D-1D Setups

We now compare all 3D-1D configurations depicted in Figure 5. After the preliminary studies in the last section, we only exchange velocity and pressure values, but no gradients. Similarly to the 3D-3D configuration, we always send velocity values from right to left and pressure values from left to right, even if we have three coupled solvers. The quasi-Newton acceleration is always computed on the 1D coupling meshes. If we have two coupling interfaces, we essentially obtain a larger quasi-Newton system, which encompasses both interfaces. The fixed-point system has a size of eight then – three velocity components and one pressure components per coupling interface. The quasi-Newton scheme shows also in these cases a very robust behavior and convergence within only two iterations in average.

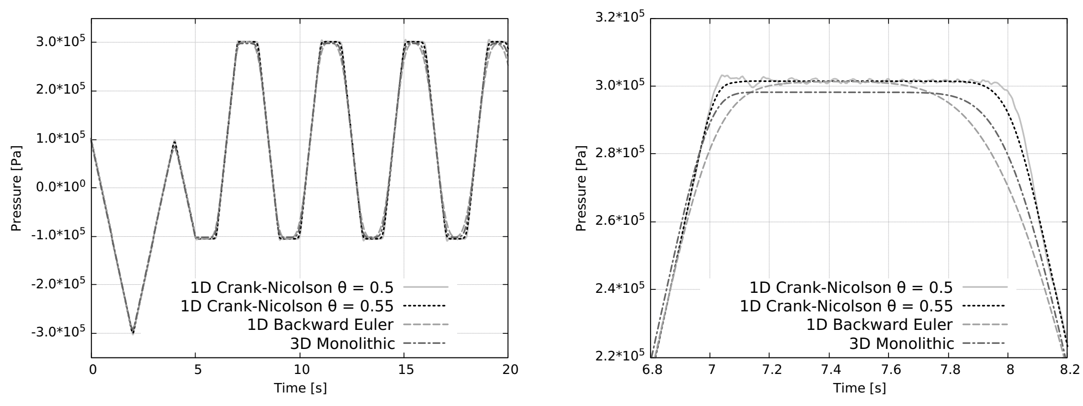


Figure 6: Comparison of timestepping schemes for the reservoir-duct-valve system (outlet pressure). Total view on the left and detail on the right. Similar results are observed for velocity.

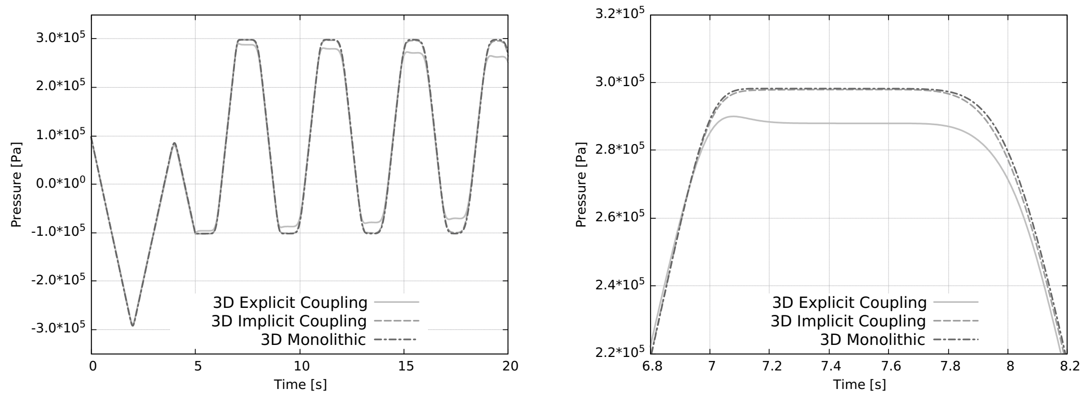


Figure 7: Comparison of explicit to implicit coupling for the reservoir-duct-valve system (outlet pressure). Total view on the left and detail on the right. Similar results are observed for velocity.

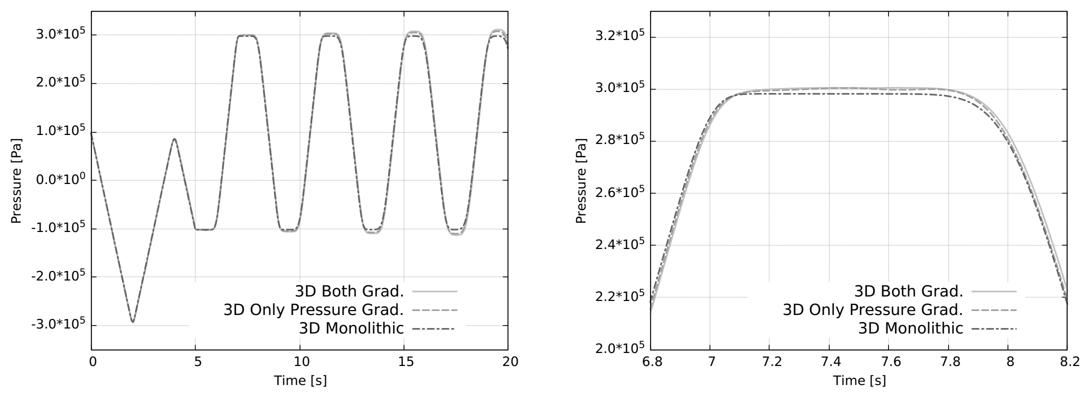


Figure 8: Comparison of different coupling conditions for the reservoir-duct-valve system. Outlet pressure over time. Total view on the left and detail on the right.

All configurations give very similar results, cf. Figure 9, which shows that the underlying physics are indeed 1D. When zooming in, we see slight differences, which match the observations of [11] fairly well. Section 4.2 already comments on the modeling error between a pure 3D and a pure 1D simulation: the 3D pressure results are slightly more damped and show a slightly lower amplitude than the 1D pressure results. For the 3D-1D configuration, we can now observe a transition between both amplitudes: first the lower amplitude, then a slight overshoot. For the 1D-3D configuration, we see the same behavior, just the other way around. Finally, for the three-field coupled configurations, the pressure waves show slightly more complicated details, but again some sort of transition.

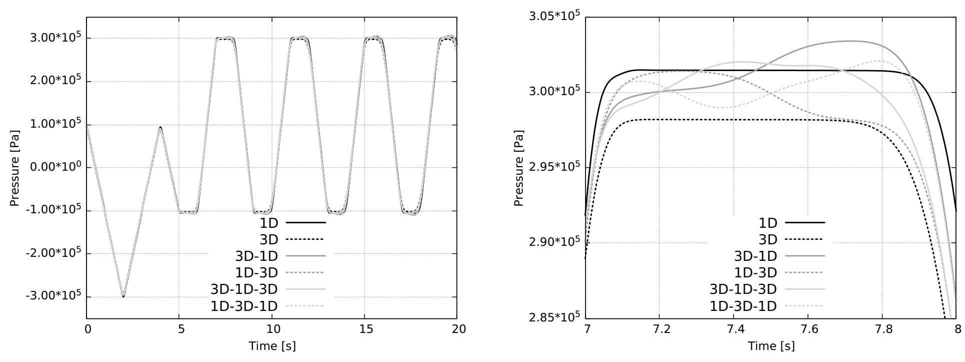


Figure 9: Comparison of all coupling configuration depicted in Figure 5 for the reservoir-duct-valve system. Outlet pressure over time. Total view on the left and detail on the right.

5 Conclusion and Outlook

Our current research aims at developing a flexible and general approach to couple OpenFOAM to any other 1D solver via the coupling library preCICE. In this paper, we start with 3D-1D fluid-fluid coupling of two pipes connected at their ends – a case we refer to as axial coupling. To achieve a truly flexible and general approach, we treat the coupled solvers as black boxes. Furthermore, we separate the geometric 3D-1D mapping, realized in preCICE, from the fluid-fluid coupling physics, realized in the OpenFOAM-preCICE adapter, an OpenFOAM function object. For both parts, prototype implementations are developed in this paper. We validate the fluid-fluid implementation in the OpenFOAM adapter by an incompressible tube flow, cut in two halfes. Afterwards, we study the water hammer effect of a reservoir-duct-valve system in detail and compare our results to those of Wang et al. [11].

For both scenarios, when only exchanging velocity and pressure values, we achieve good results validating our coupling approach in general. Exchanging velocity or pressure gradients, however, introduces a minor overshoot, which depends on the cell and time step size. We suspect a technical issue within the OpenFOAM adapter. To fully understand this behavior further study is required.

Black-box coupling seems to be a promising way to achieve the desired flexibility. We can easily couple also three (or more) arbitrary 3D or 1D components. Implicit coupling, combined with quasi-Newton acceleration, leads to an efficient and robust coupling. Two coupling iterations per timestep suffice for all tested configurations. Nevertheless, further applications and further coupled codes need to be tested to properly conclude on the generality of our approach. Next, we want to study applications in hemodynamics and nuclear reactor safety analysis. Furthermore, we plan to extend our implementation in preCICE to also radial 3D-1D coupling and the fluid-fluid module of the OpenFOAM adapter to two-phase flow and turbulence.

Acknowledgements We want to thank Håkan Nilson for the fruitful discussions that led to this work and for providing us with his OpenFOAM setups from [11]. Furthermore, we thankfully acknowledge the funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 754462 as well as funding from the German Federal Ministry for Economic Affairs and Energy (BMWi).

References

[1] H.-J. Bungartz, F. Lindner, B. Gatzhammer, et al. preCICE – a fully parallel library for multi-physics surface coupling. *Computers and Fluids*, 141:250–258, 2016.

[2] H.-J. Bungartz, F. Lindner, M. Mehl, et al. A plug-and-play coupling approach for parallel multi-field simulations. *Comp. Mechanics*, 55(6):1119–1129, 2015.

[3] G. Chourdakis. A general OpenFOAM adapter for the coupling library preCICE. Master's thesis, Dept. of Informatics, Technical University of Munich, 2017.

[4] G. Chourdakis, B. Uekermann, D. Risseeuw, et al. A general and extendible OpenFOAM-preCICE adapter for FSI and CHT applications. 2019. Article in preparation.

[5] A. de Boer, A.H. Van Zuijlen, and H. Bijl. Review of coupling methods for non-matching meshes. *Computer methods in applied mechanics and engineering*, 196(8):1515–1525, 2007.

[6] J. Herb. Coupling OpenFOAM with thermo-hydraulic simulation code ATHLET. In *9th OpenFOAM Workshop*, Zagreb, 2014.

[7] G. Montenegro and A. Onorati. A coupled 1D-multiD nonlinear simulation of I.C. engine silencers with perforates and sound-absorbing material. *SAE Int. J. Passeng. Cars - Mech. Syst.*, 2:482–494, 2009.

[8] A. Quarteroni, A. Veneziani, and C. Vergara. Geometric multiscale modeling of the cardiovascular system, between theory and practice. *Computer methods in applied mechanics and engineering*, 302:193–252, 2016.

[9] G. van Zwieten, J. van Zwieten, C. Verhoosel, et al. Nutils, 2019.

[10] V. Varduhn. *A Parallel, Multi-Resolution Framework for Handling Large Sets of Complex Data, from Exploration and Visualisation to Simulation.* Dissertation, Dept. of Informatics, Technical University of Munich, 2014.

[11] C. Wang, H. Nilsson, J. Yang, et al. 1D-3D coupling for hydraulic system transient simulations. *Computer Physics Communications*, 210:1–9, 2017.