XI International Conference on Computational Methods for Coupled Problems in Science and Engineering
COUPLED PROBLEMS 2025
A. Larese, S. Perotto and G. Rozza (Eds)

MULTISCALE AND MULTIPHYSICS SIMULATION LEVERAGING COUPLING TECHNIQUES AND STATE-OF-THE-ART CODES

SAMUELE BALDINI*, GIACOMO BARBI*, ANTONIO CERVONE*, FEDERICO GIANGOLINI*, SANDRO MANSERVISI* AND LUCIA SIROTTI*

*Laboratorio di Montecuccolino, Department of Industrial Engineering
Alma Mater Studiorum University of Bologna
via dei Colli 16, 40129 Bologna, Italy
e-mail: a.cervone@unibo.it, web page: http://www.unibo.it/

Key words: Coupling Techniques, Multiphysics Problems, Computational Methods, CFD

Abstract. The simulation of complex systems, such as the primary circuit in a nuclear reactor, requires multiscale and multiphysics capabilities due to the inherent difficulty of dealing with many physical models, geometric complexity, and time scales. A full-scale simulation of the flow geometry would require extensive computational capabilities that are not always needed, especially at the design stage of a system when the configuration can iterate quickly or simulate long pipes that are best modeled with 1-D elements. In this framework, multiscale and multiphysics techniques can significantly reduce the computational effort and offer valuable insights for research and design purposes. Single-domain computational software has been developed for a long time and offers state-of-the-art capabilities and validation for an extensive range of applications. This paper details implementing a coupling technique suitable for multiscale and multiphysics frameworks that rely on existing codes by managing the data exchange between them directly in memory. The different computational domains can be coupled via various techniques, relying on overlapping domains, boundary data exchange, or the defective boundary approach. The latest technique is the most suited to exchange data between a complex 3-D domain representing the reactor pressure vessel and the external circuit and its additional components, such as heat exchangers and pumps, that can be modeled with 1-D meshes and 0-D components. The methodology is demonstrated on some simplified models that mimic the full-scale problem to address the accuracy, robustness, and performance of the proposed approach.

1 INTRODUCTION

Multiscale and multiphysics techniques in the nuclear field arise from the challenge of accurately capturing the wide range of physical processes that govern the behavior of nuclear systems. These processes extend across very different lengths and time scales, from atomic-level interactions and microstructural evolution in fuel materials, to coolant flow and heat transfer in reactor components, and further up to full system dynamics that determine safety and performance. A purely single-scale approach typically deals only with a single domain, while some effects deriving from external models can significantly impact the simulation accuracy. For example, irradiation-induced defects at the nanoscale can influence thermal conductivity at the

fuel-rod level [1], or turbulent flow structures in the coolant channel generate disturbances in the core-wide temperature distributions [2].

To address this, multiscale methods connect simulations across scales, either by explicitly passing information upward from fine-resolution models to coarser ones, or by embedding detailed local physics into larger-scale frameworks through homogenization or surrogate models. The multiscale approach allows the construction of models that account for material degradation, crack formation, or changes in fuel morphology within reactor system analyses. At the same time, multiphysics approaches integrate the simultaneous action of distinct phenomena, be it neutronics, thermal-hydraulics, structural mechanics, and sometimes even chemistry or corrosion processes, within unified computational platforms. The interaction between these fields is particularly critical under transient or accident conditions, where the evolution of one process can strongly amplify or mitigate others [3]. Another field where the multiphysics approach can be crucial is at the design stage of innovative systems, where the feedback effect of different models can significantly modify the outcome of the single-physics computational tool.

The combination of multiscale and multiphysics modeling has become a cornerstone of modern nuclear science and engineering [4, 5, 6]. It underpins predictive capabilities for advanced reactor concepts, supports life-extension of existing nuclear power plants, and informs safety assessments. Recent advances in computational power and numerical methods have made it possible to move beyond simplified, decoupled treatments toward more integrated and predictive simulations. This shift reflects a broader ambition: to create digital frameworks that mirror the true complexity of nuclear systems, enabling deeper scientific understanding and more robust decision-making in design, operation, and regulation.

Multiscale and multiphysics techniques can be successfully applied to capture the complex behavior of fluids under extreme conditions of temperature, pressure, and radiation [7]. At the largest scale, system thermal-hydraulic (TH) codes such as RELAP5, TRACE, and ATHLET are traditionally used to describe reactor-wide transients and accident scenarios. These codes treat coolant flow with averaged, one-dimensional formulations of the conservation equations for mass, momentum, and energy. While computationally efficient, this approach cannot resolve localized phenomena such as turbulent mixing, boiling instabilities, or flow separation, which are critical in many safety-relevant scenarios.

Computational fluid dynamics (CFD) methods are introduced as a finer-scale modeling tool to address this limitation. Codes such as OpenFOAM, STAR-CCM+, ANSYS Fluent, or specialized nuclear CFD platforms solve the Navier-Stokes equations in three dimensions, often with turbulence models like Reynolds-Averaged Navier-Stokes (RANS) or Large Eddy Simulation (LES). These allow detailed resolution of velocity fields, pressure distributions, and heat transfer in coolant channels, sub-assemblies, and mixing junctions.

2 COUPLING APPROACH

Two main philosophies can be employed to implement multiscale and multiphysics techniques in thermal fluid-dynamics: developing a dedicated, unified code that integrates multiscale and multiphysics features from the ground up, or coupling together existing, well-validated codes that each specialize in a particular domain. Both approaches aim at the same goal, i.e., accurately capturing the interplay between neutronics, thermal-hydraulics, and structural behavior, but they differ significantly in methodology, flexibility, and maturity.

A dedicated code, explicitly designed for tightly integrated multiscale and multiphysics anal-

ysis, benefits from consistency in its numerical framework and data structures. This means that the discretization methods, time stepping, and solution algorithms can be harmonized from the start, reducing interface errors and often enabling tighter coupling between physics. This approach also allows developers to optimize the code for high-performance computing, taking advantage of massively parallel architectures to handle the computational intensity of DNS, LES, or coupled neutronics-thermal-hydraulics simulations. The trade-off, however, is that these codes often require decades of development to reach the level of validation and regulatory acceptance available through the use of legacy tools.

In contrast, coupling well-validated codes leverages the extensive experimental benchmarking and regulatory trust that established system TH, CFD, and neutronics tools already enjoy. This modular approach allows researchers to pick well-established, state-of-the-art software for each physics domain and combine them through carefully designed interfaces. It accelerates deployment in real-world applications because each component code has a proven pedigree, and the primary development effort focuses on the robustness and efficiency of the coupling strategy. On the other hand, challenges arise from mismatches in discretization, time integration schemes, and underlying assumptions. Achieving stable and accurate coupling across these disparate temporal and spatial scales requires sophisticated interface algorithms, and sometimes approximations that dilute the benefits of detailed physics.

Ultimately, the dedicated code approach offers a path toward seamless, future-proof integration of multiscale and multiphysics models, particularly suited for research on advanced reactors and long-term innovation. The coupled-code strategy, by contrast, is more pragmatic for near-term deployment, capitalizing on the validation heritage of legacy tools while incrementally enhancing predictive fidelity.

This paper deals with an open-source numerical library, `cocoa` (*COde COupling for Applications*), that implements the second paradigm by interfacing state-of-the-art codes most efficiently [8, 9]. Similar libraries have been developed and are currently in use, the most relevant being the `precice` library [10] and the `GenFoam` library [11], but not all the features that are required are available in this open-source software. In particular, the features that must be available include the possibility to couple different codes on interfaces that are shared between different domains, and on overlapping three-dimensional domains, as well as the possibility to use data structures that are not built in the `OpenFOAM` framework. The library is written in C++ and is fully open-source, with bindings for the python language that allow its use with a simplified interface. The library is available at `https://github.com/capitalaslash/cocoa.git`. The library has been already adopted for other application domains, e. g. in the optimal control of the heat equation [12, 13].

The library operates as a supervising agent communicating with the different codes, offering a unified software interface that allows easier reusability. Using multiple codes that interact via a coupling interface leads to a segregated solution of the different domains associated with each code. Ideally, a monolithic solution for the whole system would benefit accuracy and robustness. In some specific interactions, a tight coupling is necessary to achieve convergence, but it could not even be feasible due to its computational cost. When using segregated solutions on different codes, the coupling algorithm must provide a tight interaction between codes (inner looping) that can perform multiple iterations of the data exchange until a suitable convergence is achieved.

Figure 1 illustrates two different approaches: the Point-to-Point and the Hub-and-Spoke

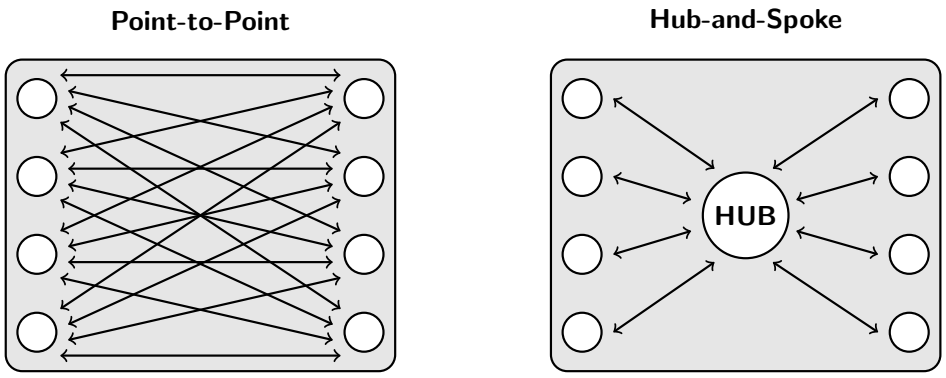

Figure 1: Point-to-Point (left) and Hub-and-Spoke (right) couling techniques.

strategies. In a Hub-and-Spoke model, all components communicate through a central hub that manages data exchange and synchronization, ensuring consistency and easier scalability. In a Point-to-Point model, components interact directly with each other, allowing flexibility but creating a complex web of interfaces. The former simplifies integration and control, while the latter risks redundancy and maintenance challenges as the number of interfacing libraries grows. The Hub-and-Spoke approach is especially advantageous for software interfaces between libraries because its common interface makes it far easier to add new libraries in the future without rewriting existing connections, supporting long-term extensibility and modular growth.

3 NUMERICAL IMPLEMENTATION

In order to exchange heterogeneous fields between different codes, it is important to unify the data structures that different codes adopt to facilitate and uniform the exchange. To this scope, the `cocoa` library uses an intermediate representation common to all codes. This representation is based on the open-source MED and MEDCoupling libraries.

These libraries have been developed in the framework of the SALOME platform. SALOME is an open-source numerical platform developed by CEA (Commissariat Ãă l'Ãl'nergie atomique et aux Ãl'nergies alternatives) and EDF (ÃL'lectricitÃl' de France) to provide open-source software for computer-aided engineering (CAE) [14].

The platform includes several modules, namely GUI, Geometry, Mesh, Fields, YACS, JobManager, and ParaViS, that can be used to manage every stage of a computational simulation performed by external standalone codes. The software implements tools for parametric CAD modeling, tetrahedral and hexahedral mesh generation, code supervision, data analysis, and postprocessing [15]. The YACS module acts as a supervisor that can manage simulation workflows that connect different computational units with the support of the FIELDS and MEDCoupling libraries. Data communication is performed by manipulating the inputs and outputs of the simulations. The module can also perform data transfer, modification, and analysis. The most relevant feature for the present work is the management of data structures and their input/output from/to files coherently to their memory representation. Furthermore, the library can perform aggregation and data exchange, interpolation between different grids, format conversion, and renumbering or partitioning of data for multiprocessor frameworks.

Specifically, the MED library is a low-level module of the SALOME platform that imple-

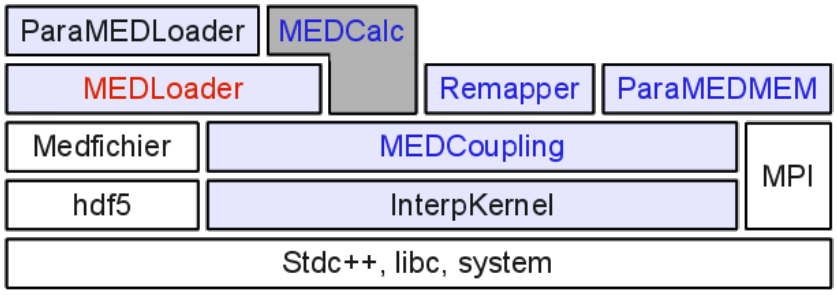

Figure 2: The architecture of the MEDCoupling platform.

ments retrieval, processing, and data sharing at the memory level, avoiding the use of external files. This library is an implementation of an abstraction layer for data structures that can be manipulated and stored on top of the HDF5 format. The MEDCoupling library is a high-level module in the SALOME platform that performs the coupling operations on the data fields. It uses MED communication for the exchange format and implements (parallel) field distribution and interpolation algorithms, buillt on overlapping and non-overlapping unfitted meshes. The structure of the library and of its dependencies is shown in Figure 2. In the following, for simplicity, we will refer to the MED and MEDCoupling libraries only by the name MED.

4 LIBRARY USAGE

The `cocoa` library provides several examples of usage, comprising overlapping domains and interface exchange, utilizing different CFD and TH libraries interchangeably.

Currently, an interface to several open-source CFD libraries has been implemented. In particular, the library can interface with the OpenFOAM library in both the Foundation and ESI Group versions. The first one is a more advanced interface, since starting from version 11, the OpenFOAM Foundation provides a class-based implementation of all the major solvers, which can be interfaced to the coupling library more easily. The ESI Group version, on the other hand, provides stand-alone applications that must be interfaced one by one with the coupling library. For this reason, only a selected number of solvers have been implemented.

The `cocoa` library also includes an interface to two in-house open-source finite element libraries, ProXPDE [16] and FEMuS [17]. These libraries implement a set of physics, including the Navier-Stokes equations, heat equation, Fluid-Structure Interaction, advanced thermal turbulence models, and two-phase flow using the VOF approach.

On the system thermal-hydraulics side, the library currently supports only a simple 1-D and 2-D finite difference solver (included in the library). The library is also capable of interfacing with the CATHARE software, which is a specialized nuclear thermal-hydraulics code that is, however, only available upon licence from CEA. Future plans include implementation of an interface to the OpenMODELICA software [18].

An example of code coupling with overlapping domains is given in Algorithm 1. The code builds two problems: a CFD problem leveraging the OpenFOAM library and a thermal-hydraulics code leveraging the internal 1-D finite differences solver. For the problem initialization, the OpenFOAM solver requires a `case_dir` that contains all the usual files required by an OpenFOAM simulation(i.e., a `0` folder with initial and boundary conditions, a `constant` folder with physics parameters such as gravity, turbulence model, etc., and a `system` folder

Algorithm 1 A one-way code coupling with overlapping domain.

```
import cocoa

problemCFD = cocoa.ProblemOForg()
problemCFD.setup(case_dir="oforg_box", config_file="oforg_box.yaml")

problemTH = cocoa.ProblemFD1D()
problemTH.setup(config_file="circuit.yaml")

coupling = cocoa.CouplingMED()
coupling.setup(problem_src=problemCFD, problem_tgt=problemTH)

while problemCFD.run() or problemTH.run():
    problemCFD.advance()
    problemCFD.solve()

    coupling.project(name="Tcfd")
    problemTH.advance()
    problemTH.solve()
```

with configuration files such as `controlDict`, `fvSolution`, and `fvSchemes`) supplemented by an additional YAML file with instructions on the required coupling. The 1-D thermal-hydraulics code includes definitions of the coupling as well. These instructions report the name of the field to be exchanged and the mesh region on which it is exchanged.

Figure 3 exemplifies the different couplings that can be performed in the example. The CFD domain overlaps the 1-D domain so that the information could be exchanged at all the shared locations between the two domains. Alternatively, the information between the CFD code and the TH code can be exchanged only in the inlet and outlet regions of the CFD domain, marked in red and blue in the figure, respectively.

The exchange could require the usage of a feedback loop, as shown in Figure 4. In fact, the exchange of information between the two codes, be it volumetric or at the boundary, can be performed only before or after the solution of the timestep. This would imply that the first code that is executed cannot see the updated solution from the second one. In order to avoid discontinuities at the exchange interface, the field that is obtained by the other solver can be interpreted as a target of a feedback loop, i.e., the field is not imposed directly in the receiving code, but instead it is integrated as in

$$\frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T = -\alpha \nabla^2 T + k(T_{\text{CFD}} - T)|_{\text{XCH}} \ , \tag{1}$$

for the energy equation. Here, T is the temperature field, $\mathbf{u}$ is the advection field, α is the thermal diffusivity, T_{CFD} is the field coming from the coupled domain, and the pedix XCH restricts the term only to the overlapping region. k is the feedback amplification constant that regulates how strongly the receiving system tries to follow the CFD system. A higher constant would produce a smaller distance between the two codes, at the expense of possible oscillations due to overshooting.

Algorithm 2 shows an example of code coupling at the interface. In this case, the coupling is performed between two CFD codes, namely ProxPDE and OpenFOAM. The coupling in this

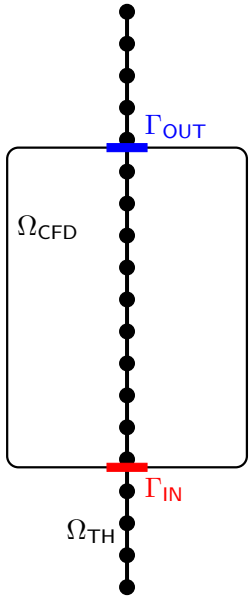

Figure 3: Coupling between a 3-D CFD code and 1-D thermal-hydraulics code.

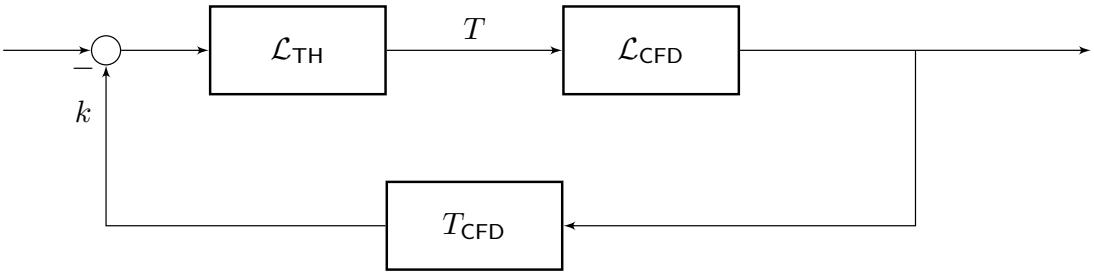

Figure 4: Feedback loop between a TH code and a CFD code.

case is two-way: both codes export a field to be used by the other code. The two domains are one after the other, so the top section of the first domain overlaps with the bottom section of the other. There are two coupling interfaces: one that communicates the fields from the first problem to the second, and the second one that performs the inverse exchange. The first problem communicates the velocity field to the second one, which uses it as an inlet boundary condition. On the other side, the second code returns the pressure distribution at its inlet, which becomes the outlet condition for the first domain. The tightness of the coupling is regulated by the `feedback` parameter, that determines how many times the step is executed to reach equilibrium between the two codes. Also in this case, a feedback control loop can be implemented to regulate how effective is a solver in following the external condition in order to achieve robustness and accuracy in the coupled domain.

5 CONCLUSIONS

The paper introduces a novel open-source library developed to perform code coupling between existing state-of-the-art codes in the field of multiscale and multiphysics simulation for the nuclear field. The importance of such tools is easily demonstrated by the requirement of simulating systems with very high complexity, leveraging codes that have been extensively val-

Algorithm 2 A two-way code coupling with an exchange surface.

```
import cocoa

p1 = cocoa.ProblemProXPDENS()
p2 = cocoa.ProblemOForg()
feedback = 3

p1.setup(config_file="box_proxpde.yaml")
p2.setup(case_dir="box", config_file="box_oforg.yaml")

coupling12 = cocoa.CouplingMED(cocoa.COUPLING_SCOPE.boundary)
coupling12.setup(
    interface_src=cocoa.CouplingInterface(p1, "top", ["vel"]),
    interface_tgt=cocoa.CouplingInterface(p2, "bottom", ["vel"]),
    method=cocoa.INTERPOLATION_METHOD.p1p1,
)
coupling21 = cocoa.CouplingMED(cocoa.COUPLING_SCOPE.boundary)
coupling21.setup(
    interface_src=cocoa.CouplingInterface(p2, "bottom", ["p"]),
    interface_tgt=cocoa.CouplingInterface(p1, "top", ["p"]),
    method=cocoa.INTERPOLATION_METHOD.p1p1,
)

while p1.run() or p2.run():
    p1.advance()
    p2.advance()

    for _ in range(feedback):
        coupling21.project("p")
        p1.solve()
        coupling12.project("vel")
        p2.solve()
```

idated for the extreme conditions that can be found in nuclear systems.

The library leverages existing open-source software and implements a generic interface that can be easily extended to other software libraries via the hub-and-spoke paradigm. Currently, the library supports many open-source CFD codes, such as OpenFOAM and the in-house finite element codes ProXPDE and FEMuS. The coupling algorithm is demonstrated in two scenarios with different features: an overlapping domain coupling between a 3-D CFD code and a 1-D thermal-hydraulic code, and an interface coupling between two CFD codes.

Future work will focus on the implementation of an interface for the generic library OpenMODELICA that can replace the usage of proprietary software for the thermal-hydraulic side, as well as on the implementation of the exchange of advanced models for turbulence and optimal control.

References

[1] Richard L Williamson et al. “Multidimensional multiphysics simulation of nuclear fuel behavior”. In: *Journal of Nuclear Materials* 423.1-3 (2012), pp. 149–163.

[2] A Ivanov et al. “Internal multi-scale multi-physics coupled system for high fidelity simulation of light water reactors”. In: *Annals of Nuclear Energy* 66 (2014), pp. 104–112.

[3] Kanglong Zhang et al. “Development of the coupled code–TRACE/TrioCFD based on ICoCo for simulation of nuclear power systems and its validation against the VVER-1000 coolant-mixing benchmark”. In: *Nuclear Engineering and Design* 362 (2020), p. 110602.

[4] Derek R Gaston et al. “Physics-based multiscale coupling for full core nuclear reactor simulation”. In: *Annals of Nuclear Energy* 84 (2015), pp. 45–54.

[5] AG Mylonakis et al. “Multi-physics and multi-scale methods used in nuclear reactor analysis”. In: *Annals of Nuclear Energy* 72 (2014), pp. 104–119.

[6] Arsen S Iskhakov and Nam T Dinh. “Review of physics-based and data-driven multiscale simulation methods for computational fluid dynamics and nuclear thermal hydraulics”. In: *arXiv preprint arXiv:2102.01159* (2021).

[7] Kanglong Zhang. “The multiscale thermal-hydraulic simulation for nuclear reactors: A classification of the coupling approaches and a review of the coupled codes”. In: *International Journal of Energy Research* 44.5 (2020), pp. 3295–3315.

[8] Giacomo Barbi et al. “Numerical Coupling between a FEM Code and the FVM Code OpenFOAM Using the MED Library”. In: *Applied Sciences* 14.9 (2024), p. 3744.

[9] Giacomo Barbi et al. “Development of a Coupling Methodology between CFD Codes”. In: *Proceedings of the 9th European Congress on Computational Methods in Applied Sciences and Engineering, Lisbon*. 2024. DOI: https://dx.doi.org/10.23967/eccomas.2024.255.

[10] G Chourdakis et al. “preCICE v2: A sustainable and user-friendly coupling library [version 2; peer review: 2 approved]”. In: *Open Research Europe* 2.51 (2022). DOI: 10.12688/openreseurope.14445.2. URL: https://doi.org/10.12688/openreseurope.14445.2.

[11] Carlo Fiorina et al. “GeN-Foam: a novel OpenFOAM® based multi-physics solver for 2D/3D transient analysis of nuclear reactors”. In: *Nuclear Engineering and Design* 294 (2015), pp. 24–37.

[12] Samuele Baldini et al. “A Finite Element–Finite Volume Code Coupling for Optimal Control Problems in Fluid Heat Transfer for Incompressible Navier–Stokes Equations”. In: *Mathematics* 13.11 (2025), p. 1701.

[13] Samuele Baldini et al. “Optimal Control of Heat Equation by Coupling FVM and FEM Codes”. In: *Mathematics* 13.2 (2025), pp. 1–24.

[14] Andre Ribes and Christian Caremoli. “Salome platform component model for numerical simulation”. In: *31st annual international computer software and applications conference (COMPSAC 2007)*. Vol. 2. IEEE. 2007, pp. 553–564.

[15] *SALOME*. https://www.salome-platform.org/?page_id=23. 2023.

[16] Antonio Cervone. *ProXPDE*. https://github.com/capitalaslash/proxpde.git [Accessed: 2025-09-26]. 2025.

[17] Giacomo Barbi et al. “FEMuS-Platform: A numerical platform for multiscale and multiphysics code coupling”. In: *9th International Conference on Computational Methods for Coupled Problems in Science and Engineering, COUPLED PROBLEMS 2021*. International Center for Numerical Methods in Engineering. 2021, pp. 1–12.

[18] Peter Fritzson et al. “The OpenModelica integrated environment for modeling, simulation, and model-based development”. In: Mic. 2022.