Growing preCICE from an as-is Coupling Library to a Sustainable, Batteries-included Ecosystem
Date: July 6, 2022
Presented by: Gerasimos Chourdakis (Technical University of Munich)
(The slides are available under "Materials from the Webinar" in the above link.)

Q. Can you compare preCICE to previous efforts in coupling libraries such as Fraunhofer's MpCCI or the coupling libraries used in the Climate community?

A. There are several alternative coupling tools, for which you can read more detailed comparisons in the introductions of our preCICE v1 and preCICE v2 papers. Both preCICE and MpCCI are addressing the same problem: coupling different codes for multi-physics simulations, both with a tradition in fluid-structure interaction and conjugate heat transfer. Both also offer integrations with existing codes, such as OpenFOAM. MpCCI is developed and distributed by the Fraunhofer SCAI institute (Germany) and is covered by a proprietary license. preCICE is openly developed on GitHub as free/open-source software (LGPLv3 license) by the University of Stuttgart and the Technical University of Munich (also Germany). MpCCI predates preCICE. On the technical side, preCICE offers some advantages. For example, while MpCCI uses a coupling server (a central component, which can limit the performance), preCICE uses a fully-parallel peer-to-peer approach, without any central component. preCICE also offers advanced acceleration schemes, such as Interface Quasi-Newton, which can greatly reduce the number of implicit coupling iterations. I do not know how different libraries used in the Climate community may be, but I am aware that there is a lot of work going on regarding coupling there as well. However, I could also see preCICE as a candidate for climate simulations, as it is a very general tool that is not restricted to fluid-structure interaction, while it offers efficient algorithms and implementation, as it has been developed in the context of the German SPPEXA project for exascale computing.

Q. Follow-on to the previous one: In climate coupling libraries there is always a balance that has to be struck between interpolation accuracy and conservation properties (important for long running simulations). How are you dealing (if you are) with this?

A. preCICE offers different space interpolation ("data mapping") methods, as well as time interpolation methods (experimental, since recently). In terms of space interpolation, the user can choose between cheap, first-order nearest-neighbor mapping, or even more accurate radial-basis functions based mapping (with both consistent and conservative variants, e.g., for temperature or force). In respect to the coupling algorithm, the user can also configure the convergence criteria of implicit coupling. Therefore, the user can decide on the methods applied, and some methods offer better accuracy than others. I am not aware of particular challenges in climate simulations, though.

Q. (Alfred Tang) The speaker makes it sound like preCICE is plug-n-play. I am still not clear what it is. Is preCICE one single package? Or a collection of adaptors for various libraries? Does it need to be compiled with all the libraries in my code? Do I need to modify my code to use preCICE?

A. The library is plug-and-play in different levels. On the one hand, the participants don't know anything about the other participant (apart from what data they are exchanging). The solvers are started normally, and only find each other at runtime. All features are also steered by the same API, meaning that a solver adapted for preCICE can configure its coupling behavior at runtime.

preCICE is a library, and one can use just the library itself, without using any of the integrations with OpenFOAM or other solvers. The library itself, together with the adapter, examples, documentation, and more components form the "preCICE ecosystem".

The library is similar to just a plotting library: one can just call it / link it from an existing code. You do need to modify your code to use the library, similarly with any other library. But your code keeps the same structure and is started normally. In the case of OpenFOAM, which offers a plugin interface, one can really build a plug-and-play simulation, without any code changes or additional compilation.

C. New reference paper "preCICE v2: A sustainable and user-friendly coupling library", in case you would like to have a look: https://open-research-europe.ec.europa.eu/articles/2-51/v1

Q. Do the coupled codes run on the same thread(s) while one is waiting for the other, or do I need to allocate more cores while submitting a job? E.g. 40 cores for CFD and 10 for FEM = 50 cores in total … That means the FEM could not use a subset of the CFD cores? Thx :)

A. No, the coupled codes start in separate processes, with separate MPI communicators. In this sense, if you want 40 cores for CFD and 10 cores for FEM, you would indeed allocate 50 cores. In the demo, I showed a serial (staggered / Gauss-Seidel) coupling scheme, in which the one participant was waiting for the other. But preCICE also offers parallel (Jacobi) coupling schemes, with which you can utilize all allocated cores, without idling at any time. If you do want to overlap the cores, you could still specify a serial coupling scheme and allocate cores accordingly (but the solvers are still running in separate processes).