

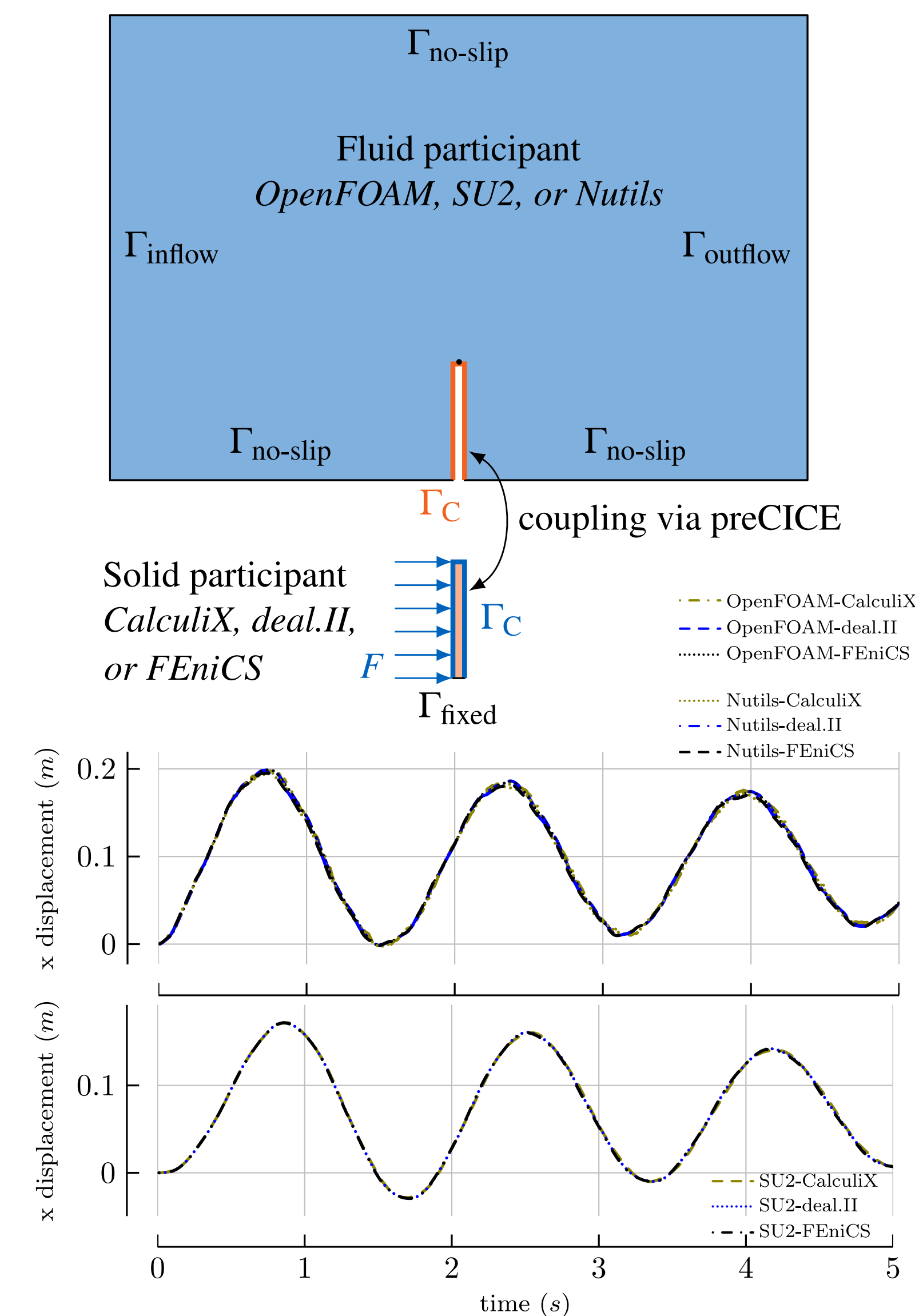
A sustainable and flexible research software ecosystem for multi-X simulations

Plain language summary

Single computer models are often not enough to describe phenomena and processes across science and engineering. The software library preCICE [1,2] allows programmers and end users to combine existing models with minimal effort, producing accurate results and being portable from a laptop to a large supercomputer. preCICE is now being sustainably developed and actively used by a vivid community spanning from aerospace and biomedical engineering to climate research. The community has now produced many components, but how can we ensure their quality and maintenance?

Example: Fluid-structure interaction

Choose your CFD (compressible / incompressible) and CSM codes at runtime: A preCICE tutorial [2].

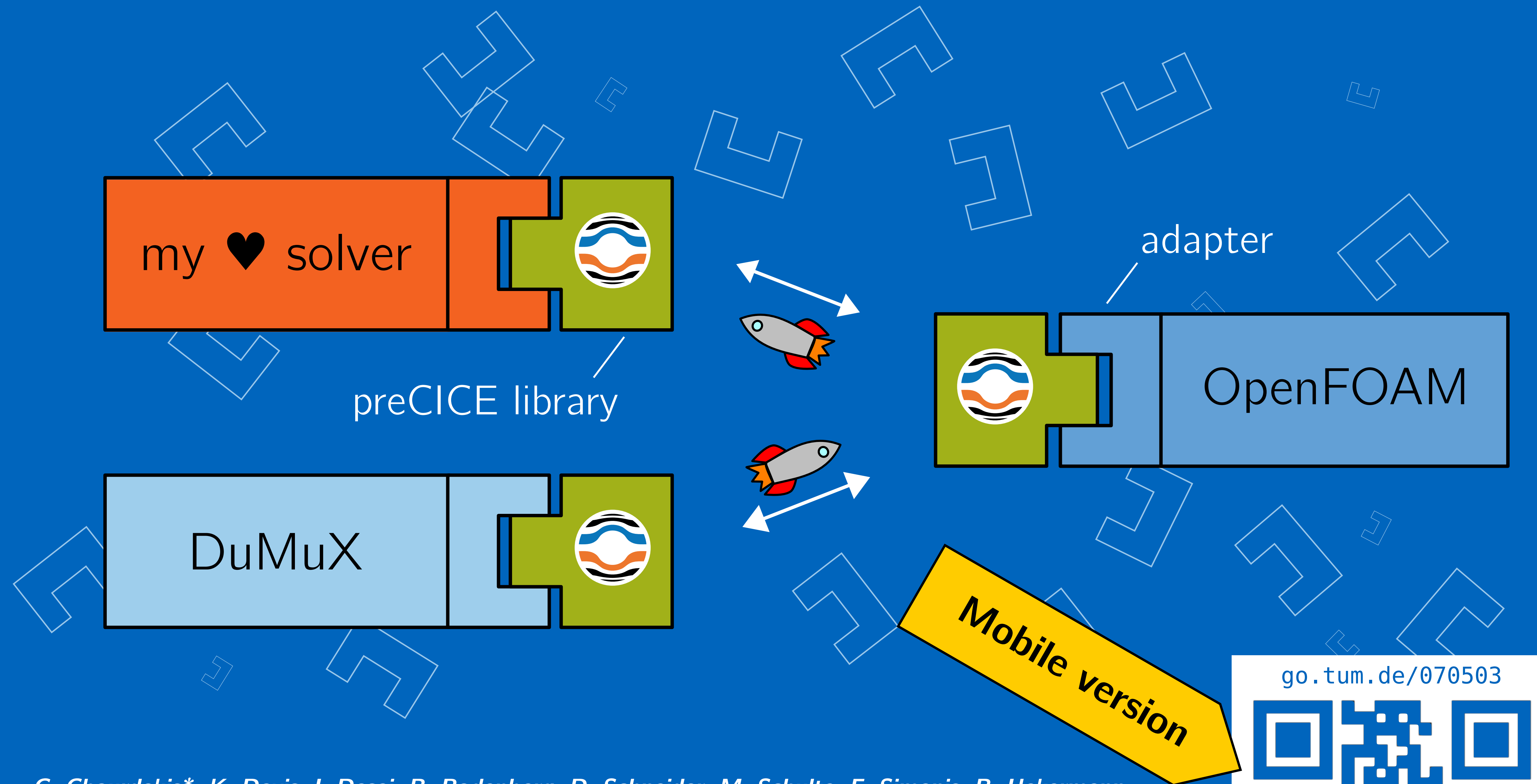


From a prototype to sustainable, easy-to-use software (with users!)

- Library approach: drop-in, no need to adopt design
- Packaging and dependencies: standard and minimal
- Ready-to-use integrations: users are not developers
- Documentation: next to the code, rendered as one
- Tests and continuous integration, code reviews
- Tutorial cases, forum, trainings, workshops

How to become FAIR? Form a community!

Facilitating a simulation coupling bazaar



G. Chourdakis*, K. Davis, I. Desai, B. Rodenberg, D. Schneider, M. Schulte, F. Simonis, B. Uekermann

gerasimos.chourdakis@tum.de

SCCS, TUM School of Computation, Information and Technology

[1] H.-J. Bungartz, F. Lindner, B. Gatzhammer, M. Mehl, K. Scheufele, A. Shukaev, B. Uekermann (2016). preCICE – A fully parallel library for multi-physics surface coupling. In *Comput. & Fluids*, 141, 250–258. <https://doi.org/10.1016/j.compfluid.2016.04.003>

[2] G. Chourdakis, K. Davis, B. Rodenberg, M. Schulte, F. Simonis, B. Uekermann, et al., preCICE v2: A sustainable and user-friendly coupling library [version 2; peer review: 2 approved] *Open Res Europe* 2022, 2:51, <https://doi.org/10.12688/openreseurope.14445.2>

[3] G. Chourdakis, K. Davis, I. Desai, B. Rodenberg, D. Schneider, F. Simonis, B. Uekermann, B. Ariguib, P. Cardiff, A. Jaust, P. Kharitenko, R. Klöforn, N. Kotarsky, B. Martin, E. Scheurer, V. Schüller, G. van Zwieten, K. Yurt (2023) preCICE Distribution Version v2211.0 <https://doi.org/10.18419/darus-3576>, DaRUS, V1.

API example (simplified)

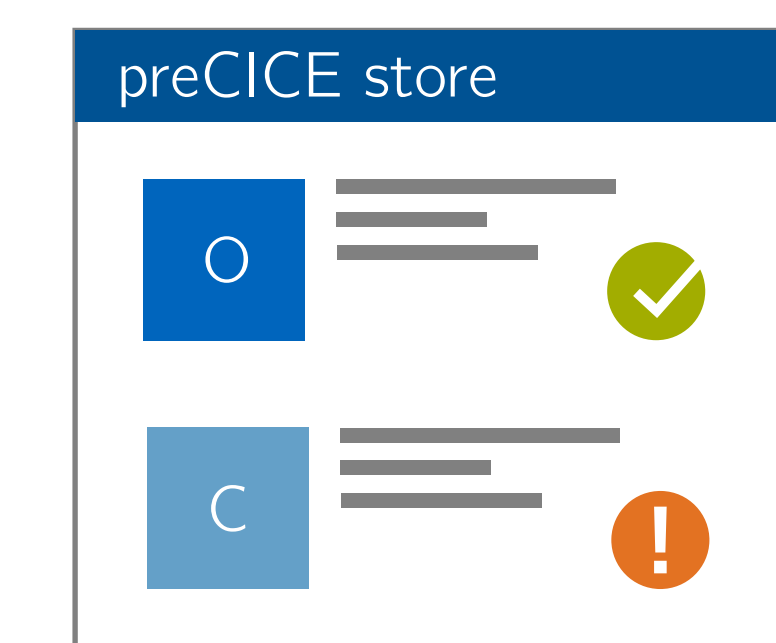
```
import precice
vertices = precice.set_mesh_vertices(positions)
precice_dt = precice.initialize()
while precice.is_coupling_ongoing():
    solve_as_usual()
    precice.write_data("force", vertices)
    precice.advance()
    precice.read_data("displacement", vertices)
```

An ecosystem of components

citable preCICE Distribution (on DaRUS [3])

Website + Documentation		Nix pkgs	VM
Tutorials			
OpenFOAM	CalculiX	FEniCS	deal.II
SU2	DUNE	DuMuX	MBDyn
Python	Julia	Matlab	Rust
preCICE (C++ core, C & Fortran bindings)			

Community-contributed components

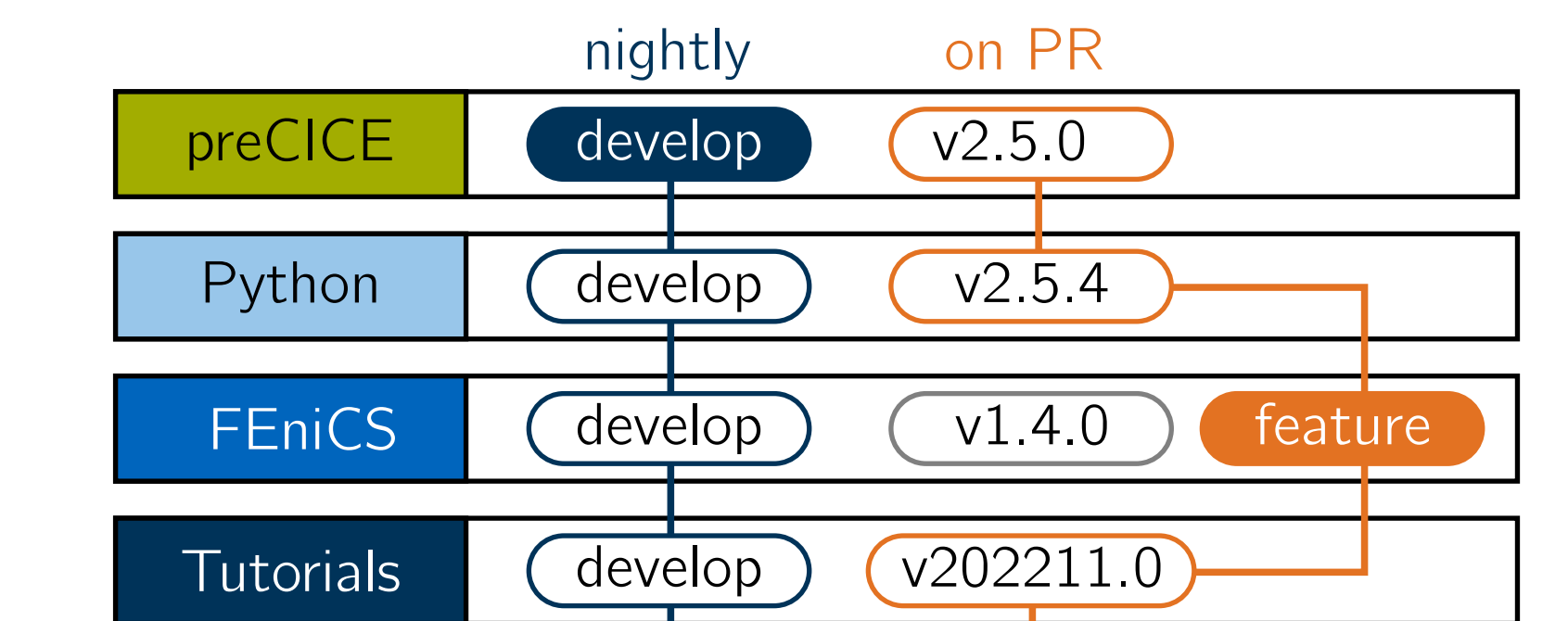


We need a standard!

- Metadata
- Tests available
- Cases validated
- Contacts, Citing, ...

Testing everything together

CI for library and some components using GitHub Actions. Partitioned approach makes CI complex.



Can we release a new version? Can we merge the PR?

Forming a community

- Go to the users (domain conferences).
- Setup public, structured community channels.
- Answer questions by editing the documentation.
- Train the users and get feedback (workshops).

