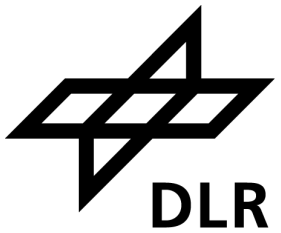
ComIn - ICON Community Interface

Aparna Devulapalli (DKRZ), Kerstin Hartung (DLR)

N.-A. Dreier[1], M. Haghighatnasab[2], P. Jöckel[3], A. Kerkweg[4], B. Kern[3], W. J. Loch[1], F. Prill[2], D. Rieger[2]
1: DKRZ, 2: DWD, 3: DLR, 4: FZJ

DKRZ

JÜLICH
Forschungszentrum



DLR

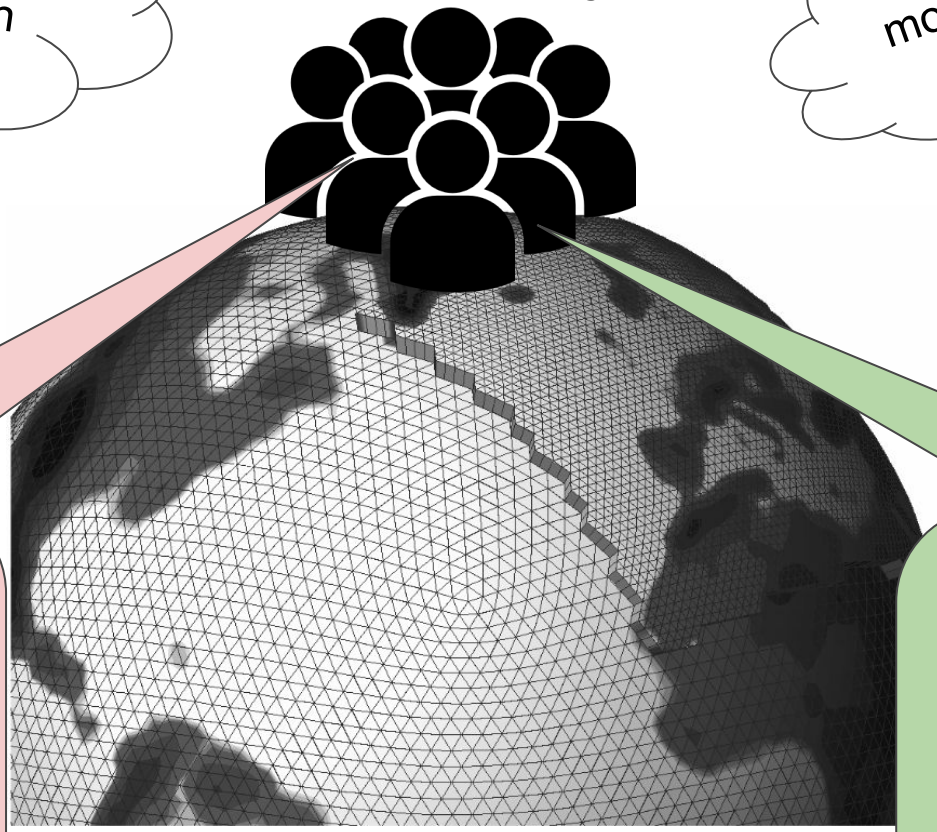
How do aerosols interact with urban pollution?

How can we better predict heat waves?

Earth System Modelling Community

What processes modulate climate change?

How does the impact of clouds change with climate change?



Development possible only for developers from ICON Consortium.

Complex scientific code requiring a steep learning curve.

ICON

Germany's primary model for weather predictions and climate studies

Stable Software Interface

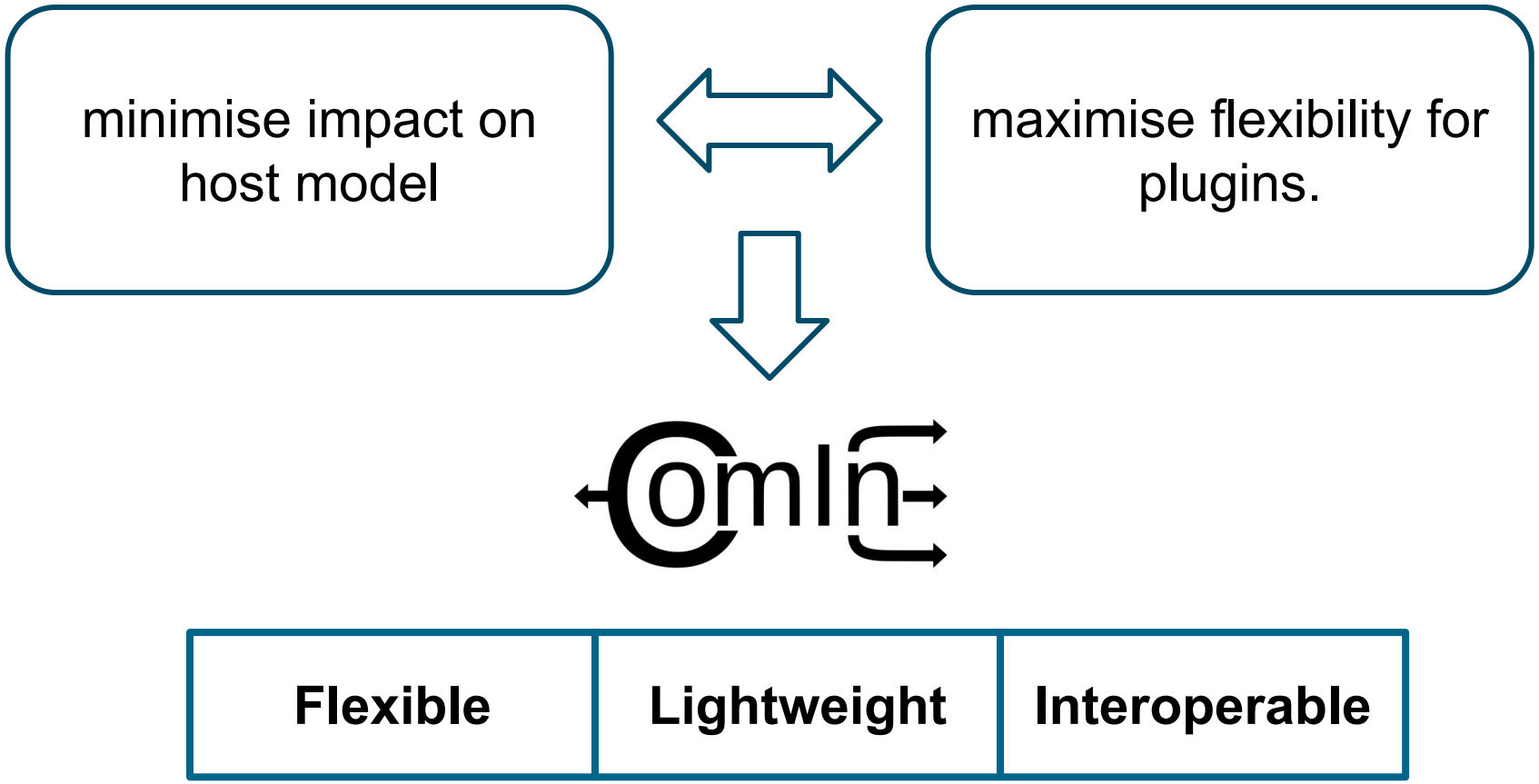
Easy Learning Curve

Ease of/Modular Access to ICON

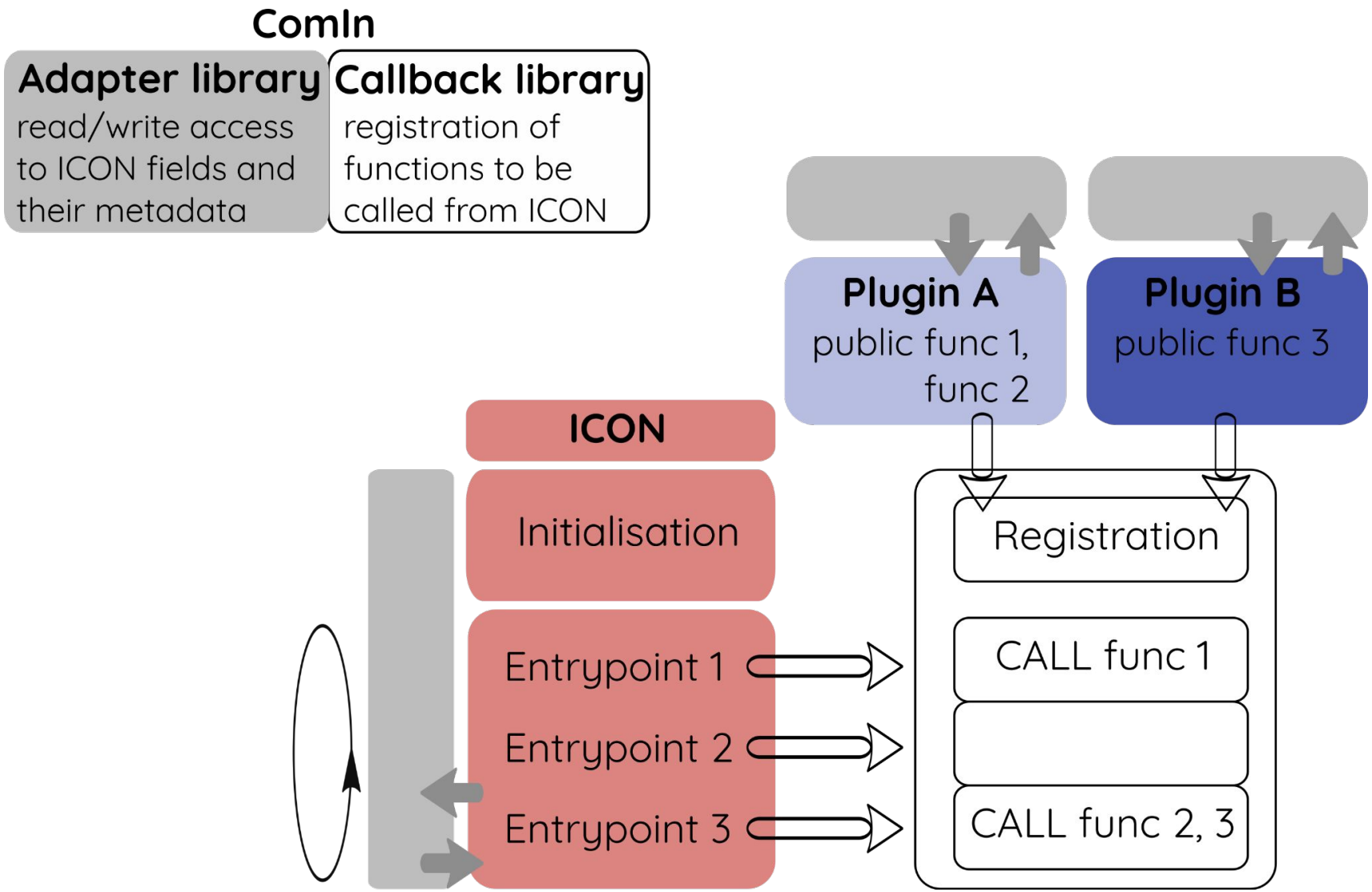
Scope for Independent development and interaction

Technical design decisions

The interface to ICON should

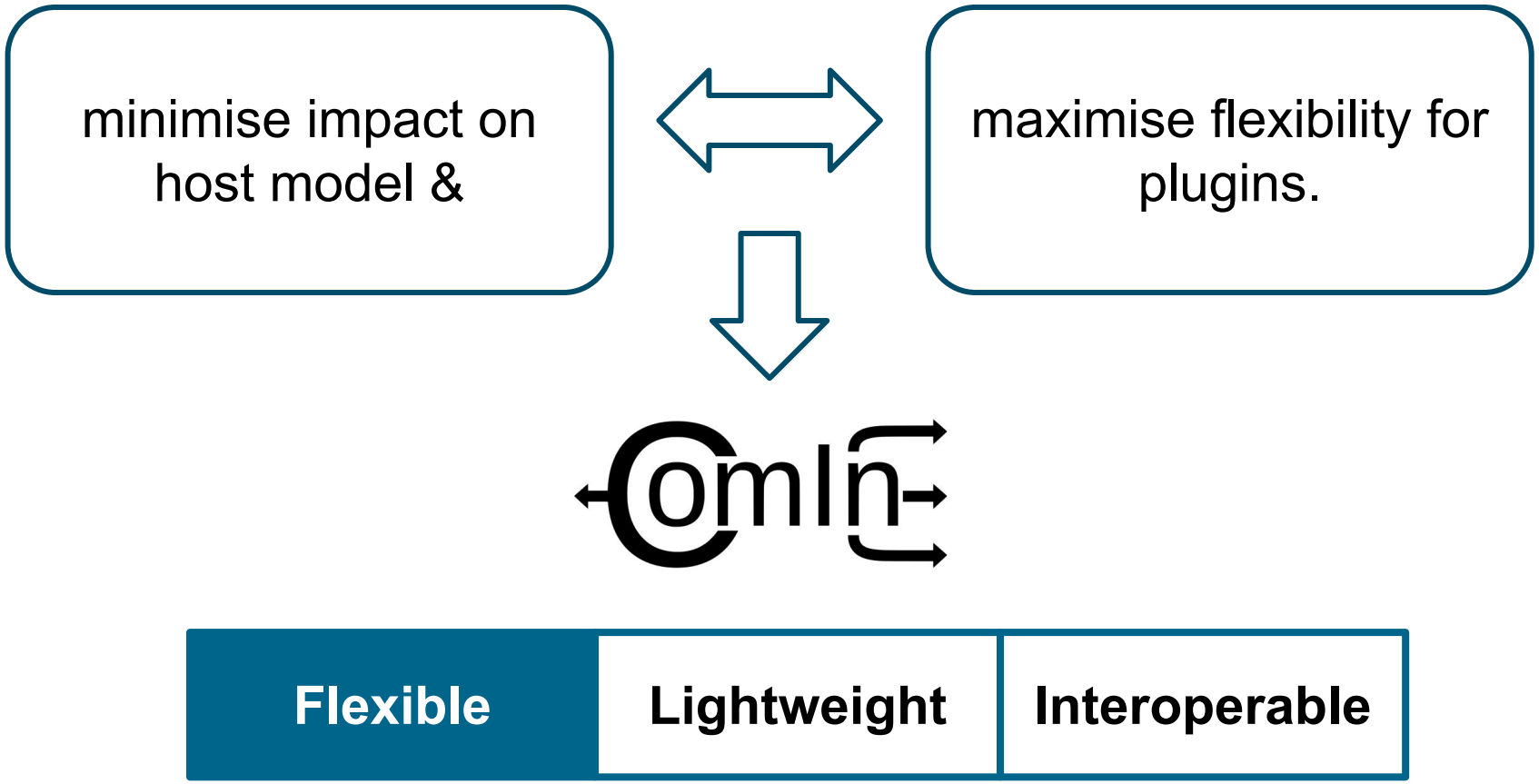


ComIn structure



Technical design decisions

The interface to ICON should

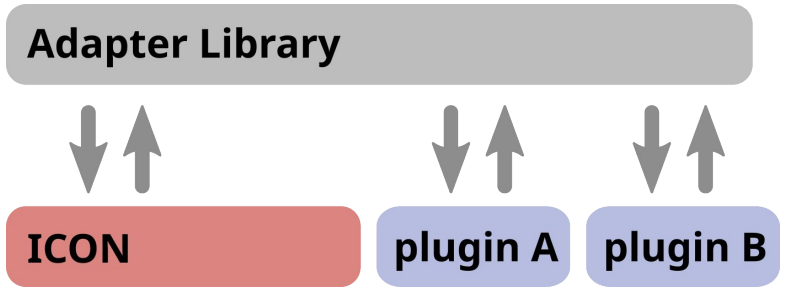


ComIn and plugins can be built and run without host model

Motivation: detach plugin development from complex host model

During the build process

- adapter library allows separate build of host+ComIn and plugin+ComIn
- plugins can be developed separately



During runtime

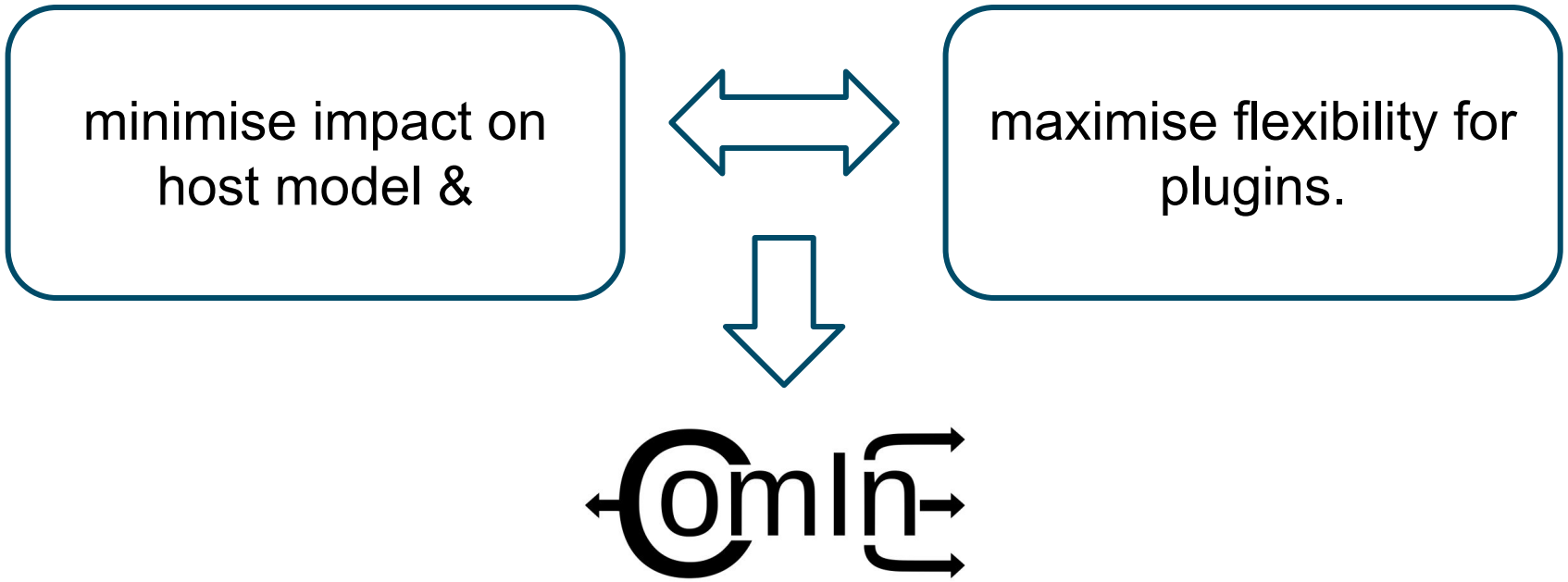
- plugin mechanism: dynamic loading, based on namelist settings
- recorder and replay tool

replay tool provides “mock” host model - adapter library content written to netCDF files

serialised data as basis for CI pipeline

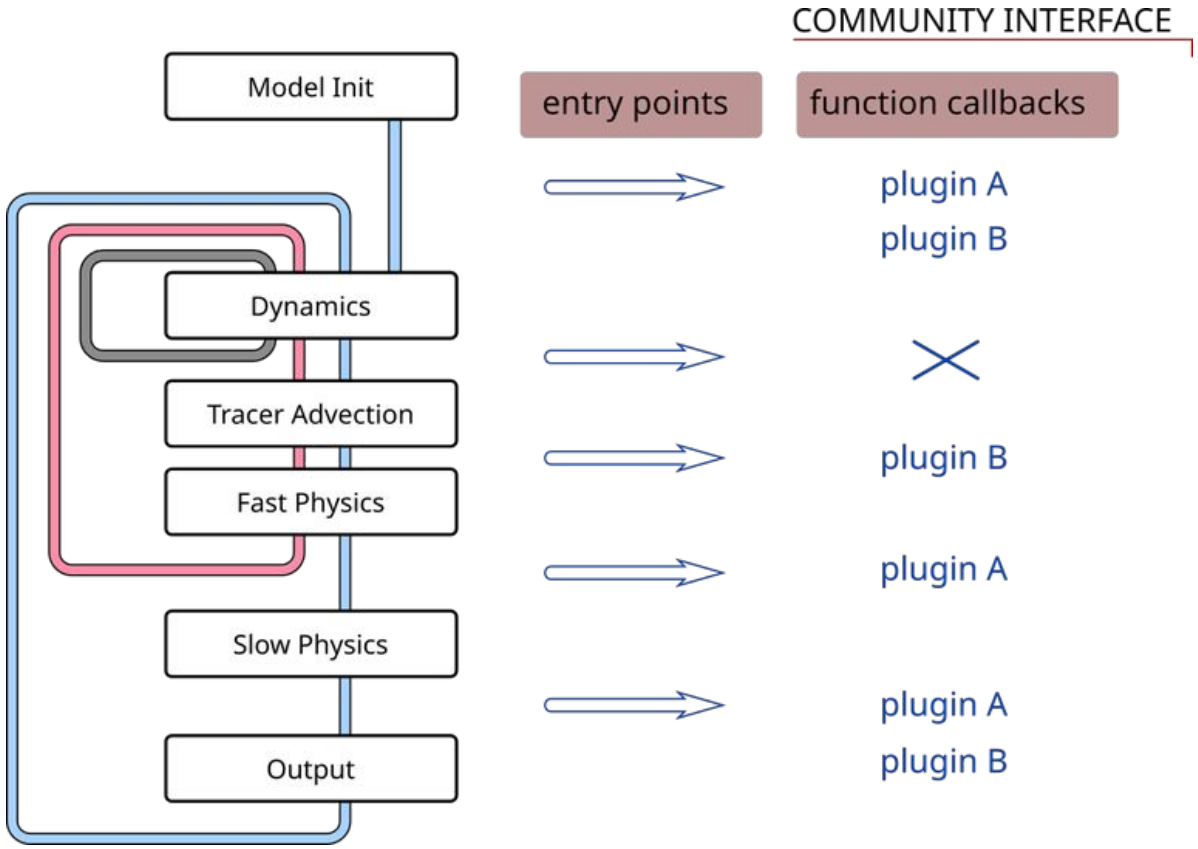
Technical design decisions

The interface to ICON should



ComIn: a data-centric event-driven library

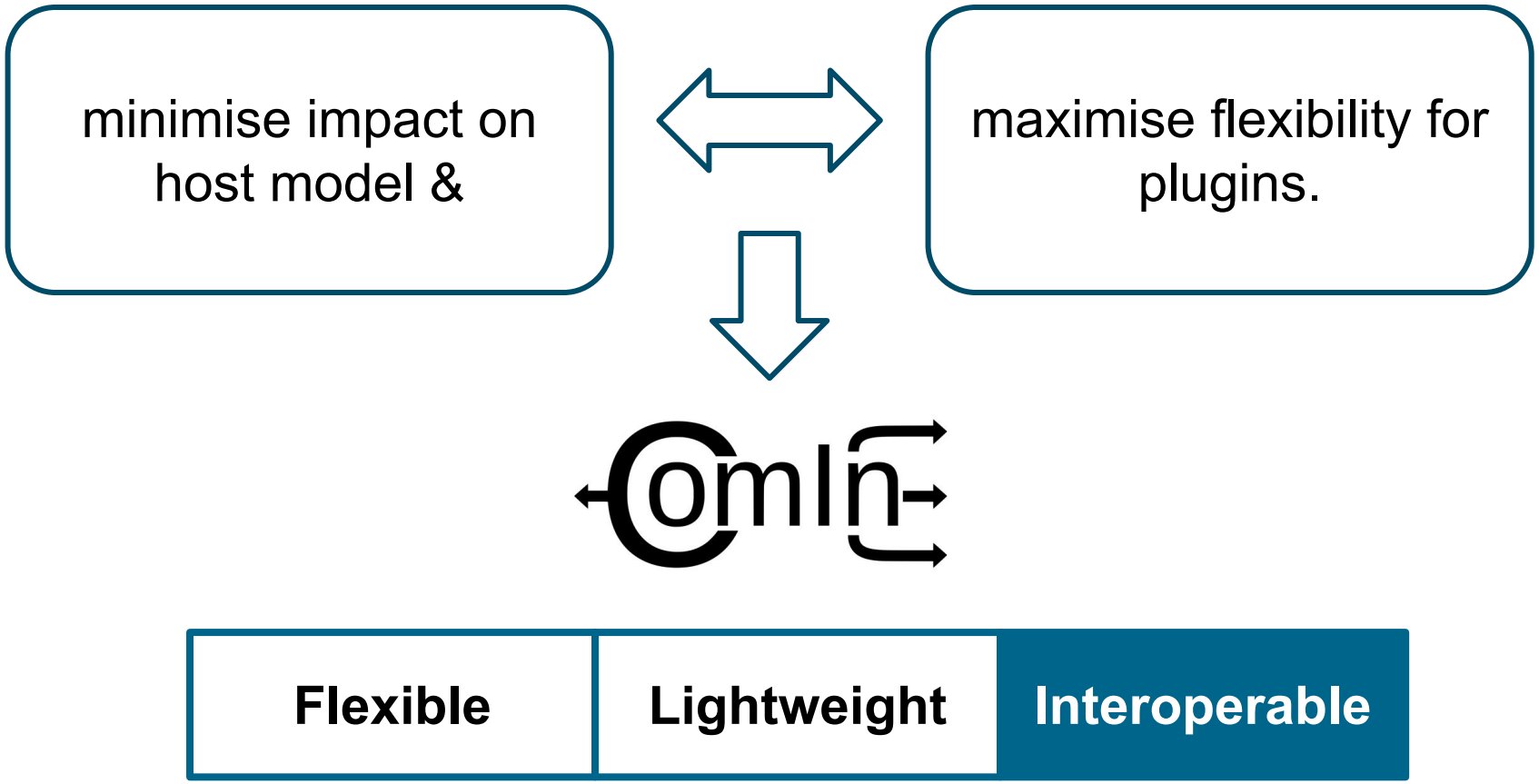
- callback and adapter library together gather information on data access patterns
- dependencies are highlighted and opportunities for parallelism revealed
- benefit compared to host model ICON, which does not provide this information



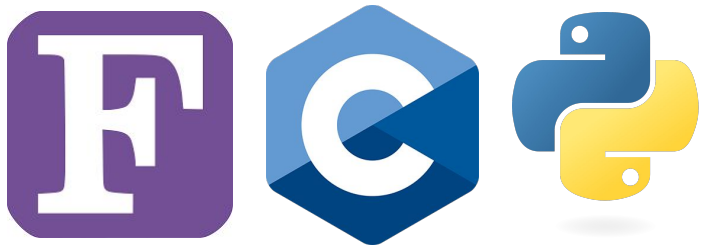
Courtesy of Florian Prill (DWD)

Technical design decisions

The interface to ICON should



ComIn offers multi-language support



Motivation: support a variety of applications

```
import comin
import numpy as np
```

primary constructor:
create and access model variable

```
comin.var_request_add(("my_var", 1), False)
```

```
@comin.EP_SECONDARY_CONSTRUCTOR
def simple_python_constructor():
    global pres, tke
    print("PYTHON: secondary constructor")
    pres = comin.var_get(
      [comin.EP_ATM_WRITE_OUTPUT_BEFORE],
      "pres", id=1)
    tke = comin.var_get(
      [comin.EP_ATM_WRITE_OUTPUT_BEFORE],
      "tke", id=1)
```

register **function callbacks** through Python decorators

```
@comin.EP_ATM_WRITE_OUTPUT_BEFORE
def simple_python_diagfct():
    print("PYHTON: diagfct called!")
    print(np.asarray(pres))
    np.asarray(tke)[:] = 42.
```

```
@comin.EP_DESTRUCTOR
def simple_python_destructor():
    print("Good bye!", file=sys.stderr)
```

See the ComIn release code for more demo plugins (Fortran, C/C++, Python).

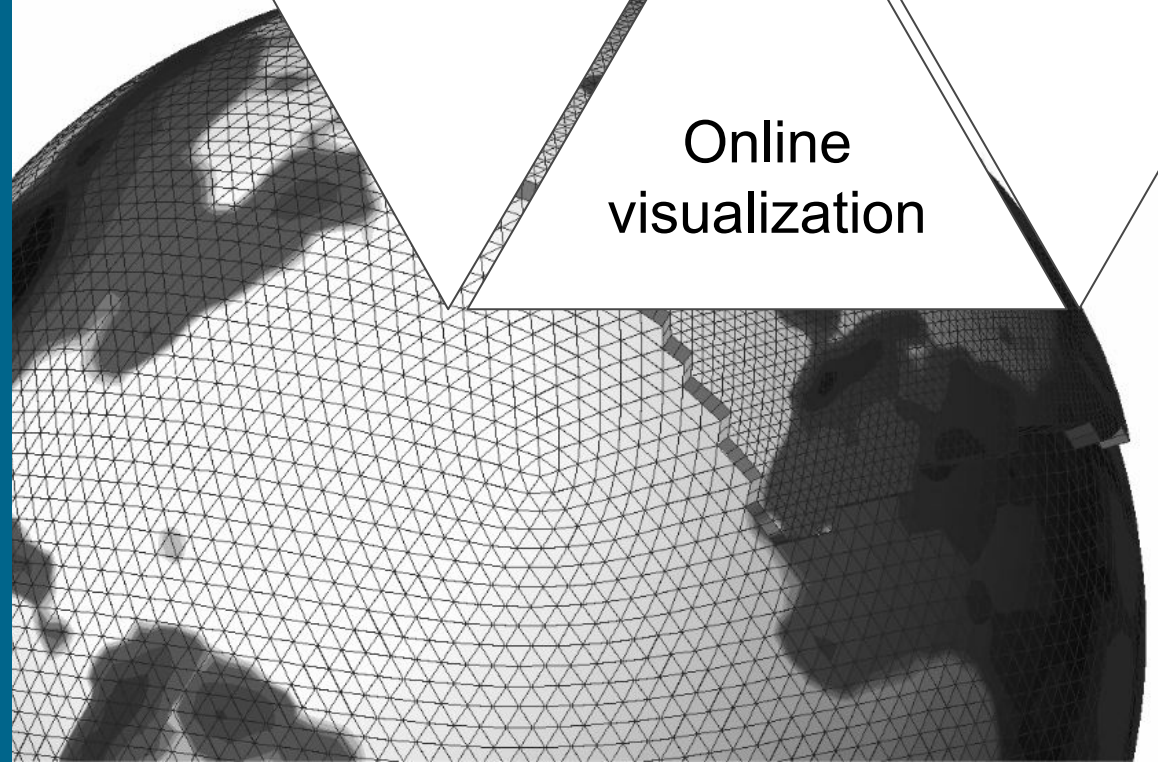
Conclusions/Take-away messages

Further plugin ideas

AI/ML

Performance monitoring

Online visualization



Accessibility for teaching

Open Science

Open Development

Further opportunities with ComIn

Scan this QR code to visit the ComIn documentation page

Thank you for listening

Question to the audience

Which API language (new or already existing)

do you prefer?

Scan this QR code to visit the ComIn documentation page

Imprint

Topic: **ComIn - ICON Community Interface**

Date: 2025-02-26

Authors: Lakshmi Aparna Devulapalli and Kerstin Hartung

Institutes: DKRZ and DLR-PA