E-CAM

European HPC Centre of Excellence

A Load-Balancing Library For Domain-Based Parallel Particle Simulations

Rene Halver

Forschungszentrum Jülich
Jülich Supercomputing Centre

Funded by the EU (Project n°676531)

Overview

- Design Philosophy
- Domain Decomposition in Particle Codes and Load Imbalance issues
- Load-Balancing Strategies
- Load-Balancing Library
- Outlook

Funded by the EU (Project n°676531)

Definition of *work*

work

scalar parameter describing the work load on a single domain, e.g. this can be:

- execution time for a single routine or a collection of different routines
- number of particles
- number of interactions

within the library it is assumed that work is distributed uniformly across a volume, i.e. the whole local domain or subsection of it, according to a uniform work distribution density

Funded by the EU (Project n°676531)

Design Philosophy

Basic ideas behind the design of the library:

- Add-on library, provides suggestion for better balanced domain borders
- No fully automatic solution for communication, but provides new neighbors for each domain
  - due to different requirements the calling program is probably better suited for communication than a general library
    - e.g. specialized data structures, domain structures, …
- Providing different load-balancing solutions

Funded by the EU (Project n°676531)

ecam

Implemented Methods

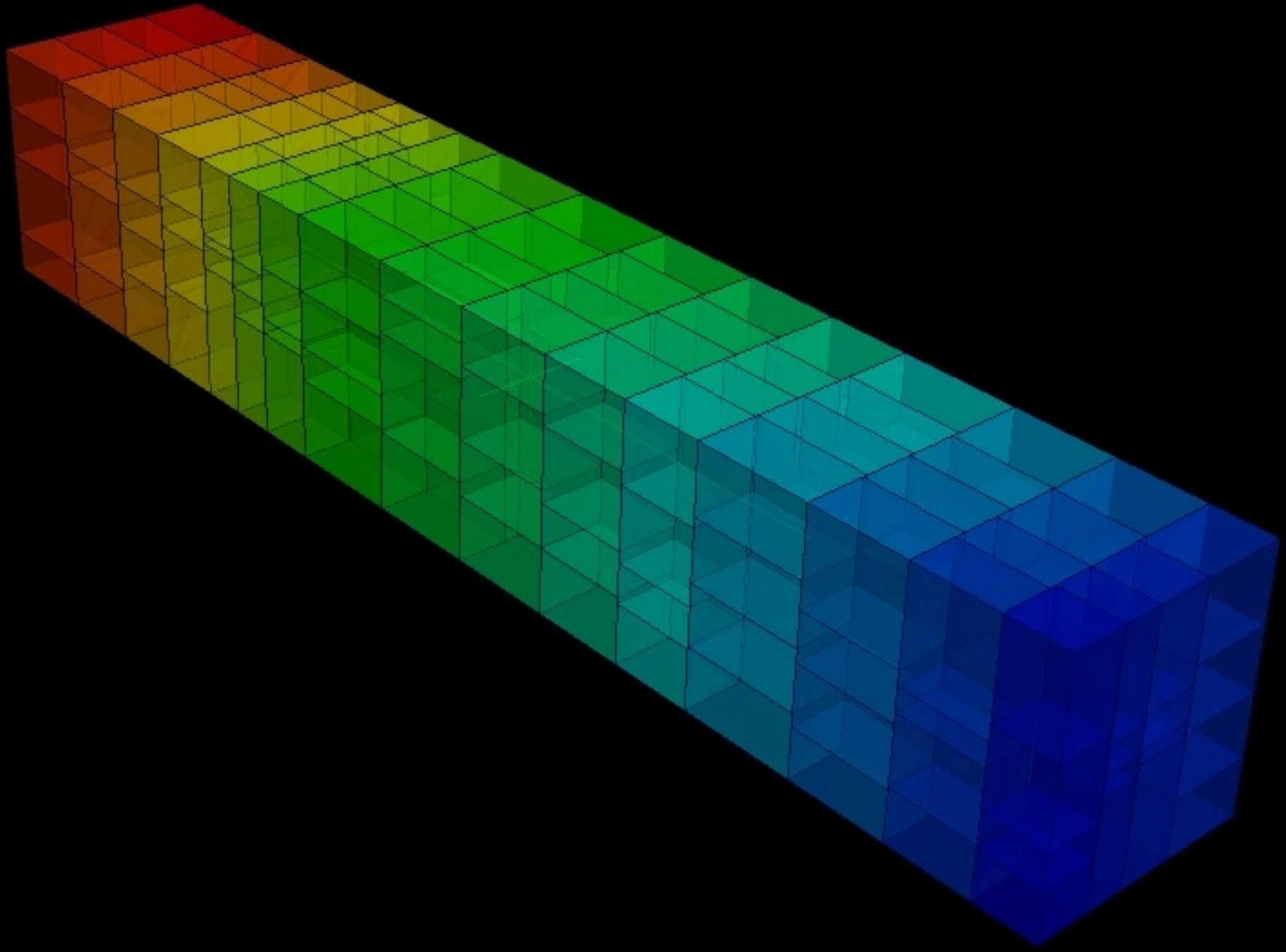
Currently implemented methods of load balancing:

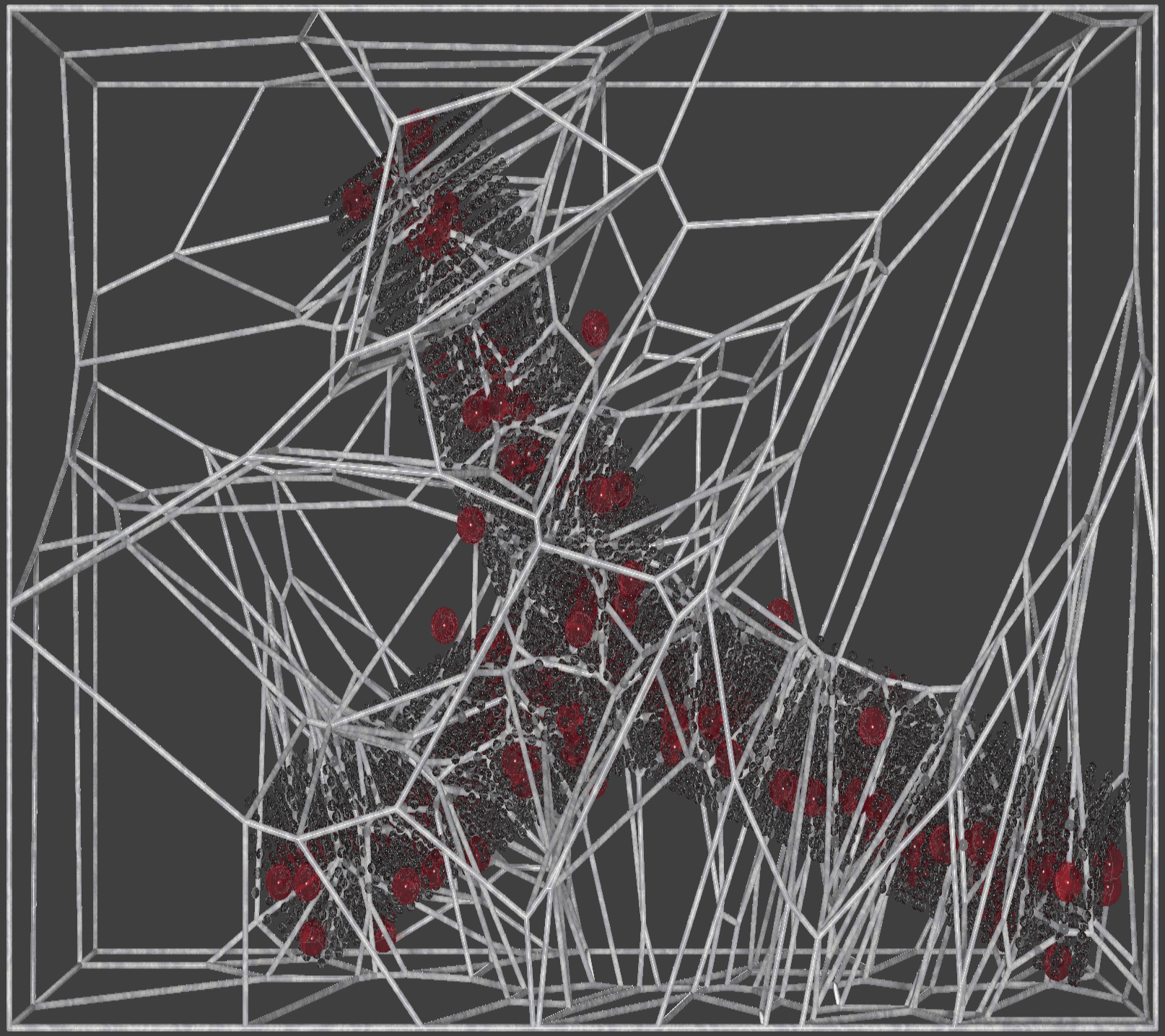
- Tensor-based
- Staggered grid
- Histogram-based

Currently worked on and partially implemented:

- Force-based
- Voronoi cell-based

Funded by the EU (Project n°676531)





Library Requirements (Calling Code)

Some requirements for the use of the library, the calling program should be able to deal with:

- shifting boundaries of domains (all methods)
- changing number of neighbors (all methods except tensor-based and force-based)
- non-orthogonal domains (force-based / Voronoi-based)

Funded by the EU (Project n°676531)

Library Requirements (Technical)

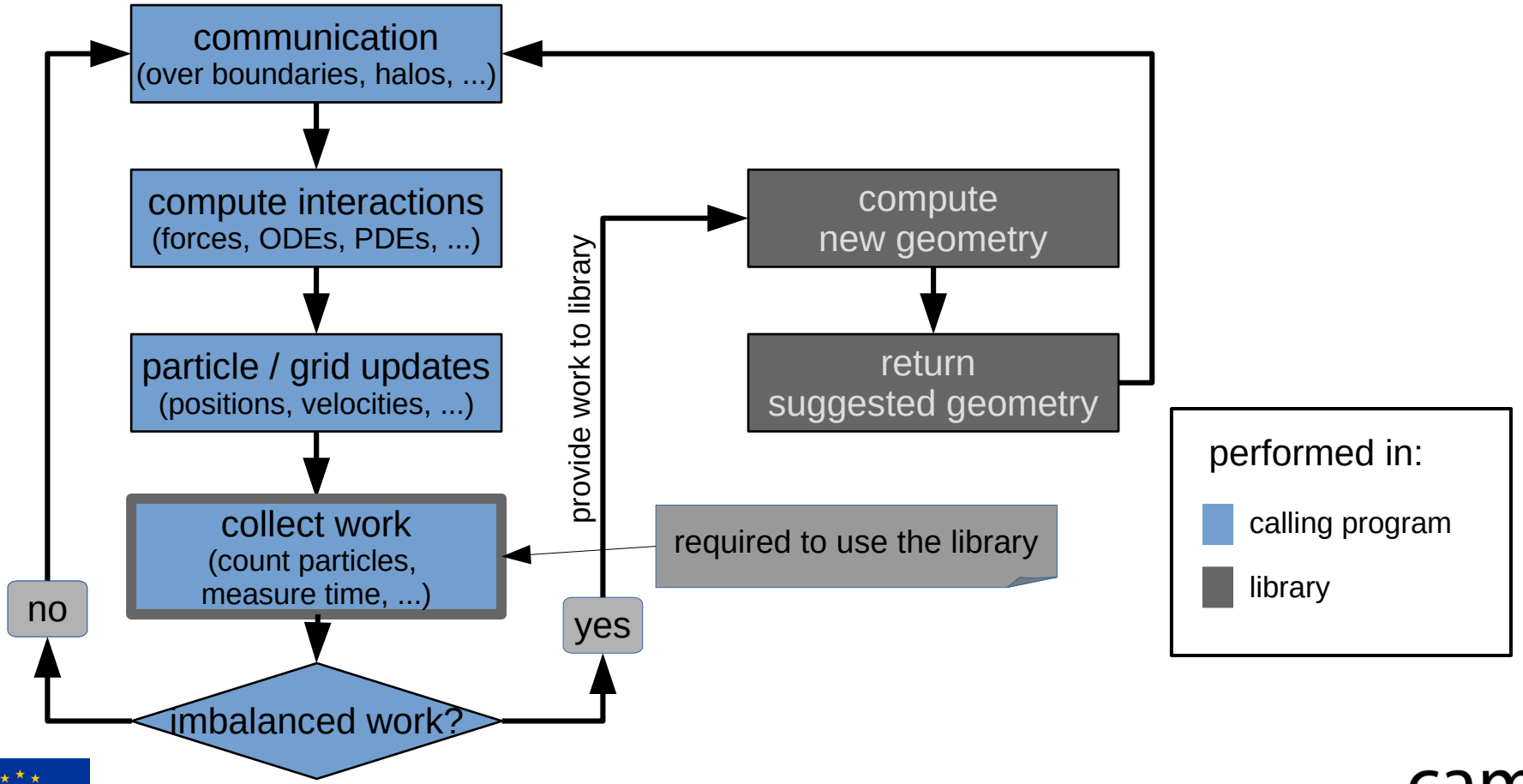
There are also some requirements on external software:

- compiler with support for
  - *C++11*
  - *Fortran 2003 (2008, if MPI module are to be used)*
- CMake 3.14+
- MPI support
- VTK 7.1+ (for visualization)
- (optional) Doxygen, Sphinx and breathe for automated documentation creation
- (optional) Boost (testing environment)

Funded by the EU (Project n°676531)

ecam

Integration into Codes



Funded by the EU (Project n°676531)

Strategies for Library Usage

There are two ways how to decide if to call the library:

- given interval of time steps / execution time

  define an interval after which the library is called, independent of current state of work balance

- test load-balance quality

  define a metric with which you describe the quality of load balance and check if the load balance quality is below a given threshold, if it is, run the library

Funded by the EU (Project n°676531)

Library Calls at Regular Intervals

advantages:

- does not require to check the quality of the load balance regularly
- provides a defined pattern of calls (e.g. once every *X* time steps)

disadvantage:

- calls the library even for already well balanced systems
  - *might lead to domain shifts and particle transfers that do not decisively impact load balancing quality*

Funded by the EU (Project n°676531)

ecam

Triggered Library Executions

advantages:

- only calls the library if required (especially useful for expensive methods, like histogram-based)

disadvantage:

- requires regular (global) checks of load-balance quality

Funded by the EU (Project n°676531)

Improving Call Efficiency

current work in progress: a function that suggests, if the library should be called / should compute new boundaries, depending on previous calls / state of system

could be combined with regular intervals in which this function is called to lessen the computational overhead for triggered call checks

Funded by the EU (Project n°676531)

Library Input and Output

required input:

- local work
- local boundaries / vertices
- MPI communicator (and process grid information, if not Cartesian communicator)

provided output:

- suggested local boundaries
- updated list of local neighbors

} suggested new domain decomposition

Funded by the EU (Project n°676531)

ecam

Other Details

- library is written in C++ and provides Fortran interface
- open source under BSD-3 license
- using Gitlab and Gitlab CI features
- released as version 0.9 (as of December 2020)
- library website:

  http://slms.pages.jsc.fz-juelich.de/websites/all-website/

Funded by the EU (Project n°676531)

Interfaces

C++ codes:

- library consists of headers → no linking required
- library provides a central class (ALL) that takes required parameters and provides new domain boundaries

Fortran codes:

- Fortran 2003 based version, using derived types or function calls
- optional: support for Fortran 2008 MPI module (mpi_f08)

Funded by the EU (Project n°676531)

ecam

Current State of the Library

the library currently provides:

- several functions to provide new domain geometry / topology
  - new domain boundaries / vertices
  - new domain neighbors
- output of domain geometry in VTK format for visualizations
- allows to maintain a minimum domain size
- small code examples to demonstrate the usage of the library

Funded by the EU (Project n°676531)

ecam

Outlook

Planned future developments include:

- new load-balancing methods, e.g. bisection-based
- automatized recognition of load-balance quality
  - possible use to automate the determination if the library should be called or not
- finalizing force- and Voronoi-based methods
- support for non-strict staggered Cartesian grids (which cannot be described by $n_x \cdot n_y \cdot n_z$)

Funded by the EU (Project n°676531)

ecam