EXTENDING PEANO4: COUPLING SOLVERS FOR SYSTEMS OF PDES

Dmitry Nikolaenko

Durham University

Introduction

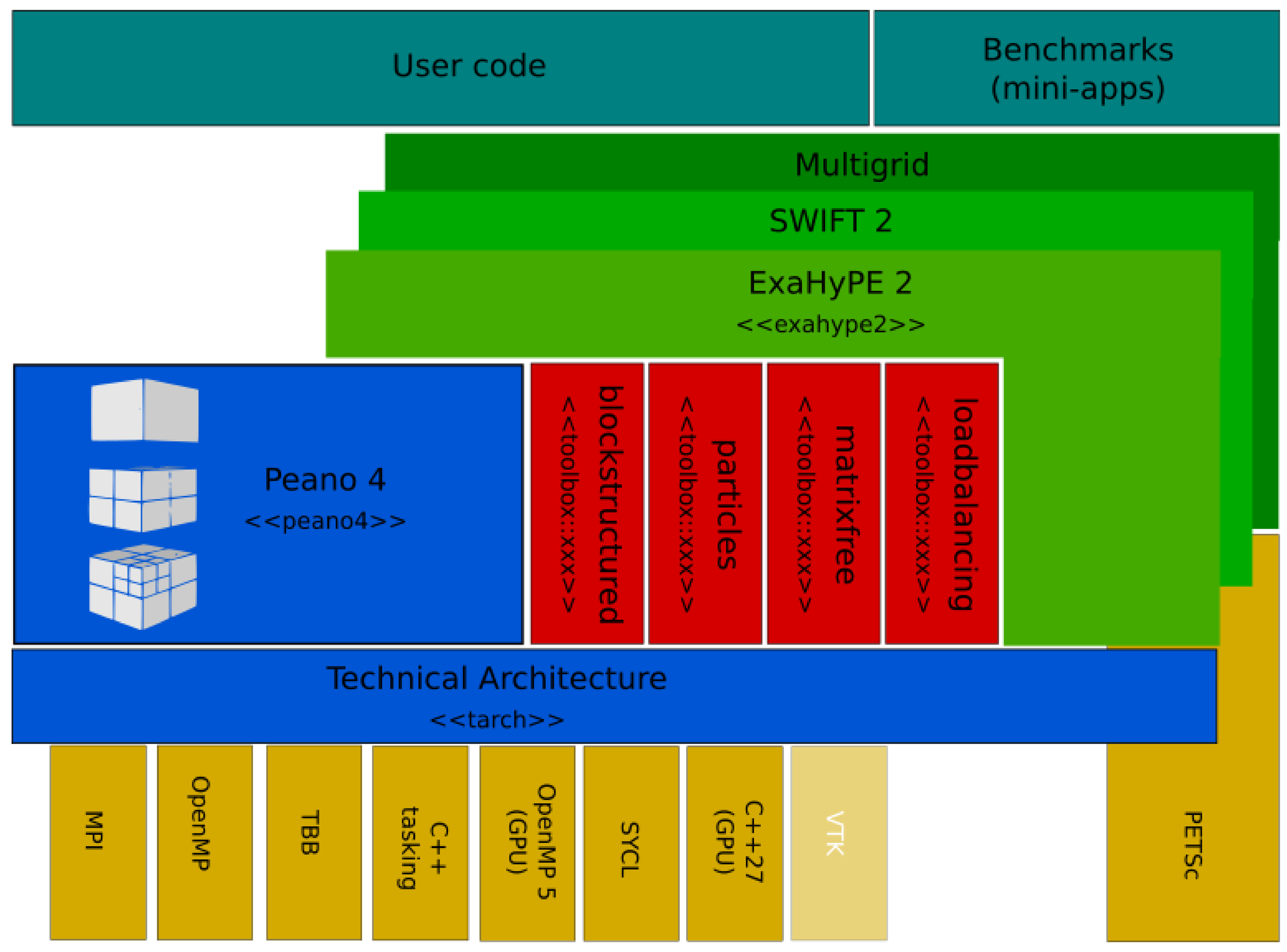
Peano is a massively parallel PDE solver with hierarchical meshes and adaptive mesh refinement [2]. Peano provides a *framework* for dynamically *adaptive Cartesian meshes* which are traversed based on the Peano *space-filling curve*. Peano is parallelised and capable of exploiting the full resources of supercomputers. Peano4 (in its current version 4) is shipped with several *specialised extensions* following Peano4 architecture as well as application and benchmarks demonstration work.

ExaHyPE2 is one of Peano4 extensions for solving systems of first-order hyperbolic partial differential equations (PDEs) [1] (e.g., in problems of seismology and astrophysics). It provides a generic engine and collection of solvers (such as finite volume method and higher order ADER discontinuous Galerkin schemes) to solve systems of PDEs.

Multigrid is another Peano4 extension being developed in collaboration with mathematicians from Bath University. It implements elliptic solvers based on multigrid methods using a hierarchy of discretisations.

Every Peano4 application, including ones based on the above extensions, is a C++ code which follows the unified Peano4 architecture consisting of several layers:

- Technical Architecture «`tarch`»
- Peano4 core «peano4»
- Various toolboxes, on top of which extensions are built, such as:
  - «`toolbox::blockstructured`» for blockstructured meshes
  - «`toolbox::particles`» for particle management
  - «`toolbox::loadbalancing`» for dynamic load balancing



Motivation

The peano4 core is written in C++, but an extensive Python API simplifies creating applications by writing a Python script generating all C++ code constituting a Peano4 application. I pick up one extension (*ExaHyPE*, *Multgrid*, etc.) and write my application using the extension's API. I'm interested to use API from several extensions.

Example problem formulation

Hyperbolic problem for Euler equation:
«Euler system of PDEs»

Dirichlet problem for Poisson equation:

$$\begin{cases} \Delta u = -f(x,y) & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega, \end{cases}$$

General workflow

(using an example of a Peano application based on *Multgrid* extension):

1. Create an `mghype` Project
2. Construct matrices
3. Instantiate solvers and add them to the Project
4. Configure the Project
5. Generate a peano4 Project

Problem

Implement a coupling between a hyperbolic solver (provided by *ExaHyPE*) for the Euler equation and an elliptic solver (provided by *Multigrid*) for the Poisson equation.

Solution

1. Create the 1st *Peano* application from one extension, e.g. *Multigrid*, following the same workflow as above
2. Create the 2st *Peano* application from another extension, e.g. *ExaHyPE* in a similar fasion
3. Merge the above 2 peano4 Projects (the new feature of *Peano4* API)

Discussion

In line with the theme proposed for RSECon24, My RSE work tries to follow some guiding principles of FAIR for Research Software:

- My implementation enables new ways of coupling solvers for different PDE systems which weren't possible before, thus improving *interoperability*. Various Peano4 extensions, such as *ExaHyPE* and *Multigrid* become capable of data exchange through API.
- This in turn makes easier *reusability* of the extensions as building blocks.

Demonstration/Results

Preliminary results
...
...
...
...

Discussion

In line with the theme proposed for RSECon24, My RSE work tries to follow some guiding principles of FAIR for Research Software:

- My implementation enables new ways of coupling solvers for different PDE systems which weren't possible before, thus improving *interoperability*. Various Peano4 extensions, such as *ExaHyPE* and *Multigrid* become capable of data exchange through API.
- This in turn makes easier *reusability* of the extensions as building blocks.

?

References

[1] Exahype: An engine for parallel dynamically
Physics Communications, 254, 2020.

[2] T. Weinzierl. The peano software—parallel, aut
TOMS, 45(2), 2019.

Contact:
Dmitry Nikolaenko
dmitry.nikolaenkodurham.ac.uk