CORSIKA8 development

Framework structure and design

A. Augusto Alves Jr

Presented at CORSIKA Cosmic Ray Simulation Workshop - KIT, Karlshuhe
June 22 2020

Karlsruhe Institute of Technology

Outline

- Overview of CORSIKA8 design guidelines
- Framework structure, building and distribution
- Dependencies management
- Ongoing code refactory
- Ongoing research topics: stack data structures, concurrency-friendly RNG and logging system

CORSIKA8 design guidelines

The CORSIKA8 development guidelines are lengthily outlined in R. Engel et al., Comput. Softw. Big Sci. 3 (2019) 2. Here is a short summary:

Code. C++17 compliant. Name identifiers, classes and functions to make clear the developer's intentions and code functionality. Favor static over dynamic polymorphism, always avoiding direct template instantiation.

Documentation. Inline commenting are our friends: *"Doxygenable"* for interfaces and normal ones for implementations. Developer's and user's manual.

Computing. Efficiency and scalability for deployment in HPC environments, which translates to multiprocess, multithread and multiplatform friendliness. Precision and reproducibility.

Test-driven development. Specification and testing of interfaces and routines using test cases. Continuous building and code coverage assessment.

Physics validation suite. To achieve the highest quality of simulations on extended timescales.

Components and their representation

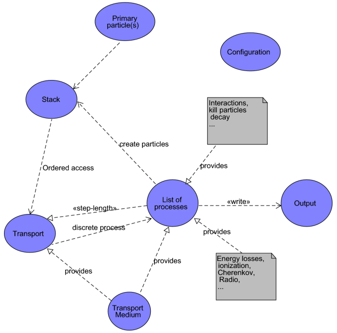
The figure at left illustrates the main components involved in the particle transport calculations. These components are represented in CORSIKA8 by different entities:

Data files. Consists data and metadata, storing necessary information for simulation: particle's features , magnetic field maps, media description, processes details and so on.

Configuration files. Storing top level directives for configuring the whole simulation job.

Data stuctures. Transient particle stack, output data container, logger sinks and output streams.

Algorithms. RNGs, particle decayers, trackers, solvers, geometry handlers, processes management and so on.



Huge combinatorial complexity if ones considers the different file formats, types of data structures and algorithms involved.

Particle cascade simulation

The double loop in the code snippet below summarizes the process the full particle cascade simulation:

- The double loop will eventually get evaluated due termination conditions: empty stack, energy cuts, environment geometrical limits reached etc.
- Simulation output can be generated in inner and outer loop.
- Many of the components of the framework will be invoked.

```
while(!stack.IsEmpty()){
    while(!stack.IsEmpty()){
        auto particle = stack.GetNextParticle();
        //particle transport
        Step(particle);
    }
    //compute evolution of particle densities
    //as they propagate through a medium
    cascadeEquations.Solve();
}
```

The parallel evaluation of this double loop requires careful analysis: potential race condition in the interaction with the particle stack, too many different algorithms involved, concurrent output generation.

CORSIKA8 framework structure

CORSIKA8 framework code is organized in core and associated modules:

- Particle stack implementation, main loop logic, units and geometry handling are part of core facilities.
- Other modules, not necessarily implemented inside CORSIKA8, provides the additional functionality.

The modular design provides a great deal of flexibility and allow users to plugin new functionality to extend or customize CORSIKA8's features.

Dependencies management

The usage of external code and facilities is unavoidable and potentially places pressure over the development and maintenance cycle of CORSIKA8. The adoption of dependencies needs to be regulated by careful analysis, based in criteria like:

- Impact of the dependency in the overall performance of CORSIKA8 should be minimal. Dependencies involved in core functionality implementation or in hot path calculations are good candidates for reimplementation.
- Multithread friendliness of dependencies need to be considered to avoid penalties or conflicts with the overall design.
- Development cycle, stability and quality of dependencies should be always considered.
- Used functionality needs to be always compared to dependency weight before to decide for reimplementation.
- Licensing compatibility.

Refactory

In order to enforce the conformity with the development guidelines and avoid future potential problems related to early choices regarding the overall framework distribution, building and maintainability, the CORSIKA8 development team decided to perform a code refactory. The main goals are:

- Make the source tree hierarchy clear.
- Enforce one file for one class convention.
- Separate interfaces from implementation details.
- Enforce coding conventions.
- Correct many other minor inconsistencies.

Refactory: first round

Focused on the source tree structure:

- All `C++` headers now have extension `.hpp`.
- Headers containing only declarations now. DOXYGEN documentation to be stored only in the headers.
- Implementations moved to `.inl` files, which are post included. No DOXYGEN documentation here, only normal comments for developers.
- Include guards eliminated and adoption of `#pragma once`.
- All test cases moved into a dedicated `tests` folder.

Re-factory: first round

On-going tasks:

- Define `namespace detail` to hide implementation details.
- Get ride of some namespaces and directories in order to fully mirror the directories hierarchy at the namespace hierarchy.
- Make all classes “first class”.
- Analyze all classes for resource management (RAII).
- Identification of design patterns.

Other topics

- Designing and developing a logger interface suitable for CORSIKA8.
- Researching on concurrently data structures suitable for a efficient data-centric design for the particle stack.
- Investigating a thread-safe, efficient and concurrency-friendly RNG based on RANLUX.

Summary and comments

- Many opportunities to contribute.
- Follow the work at: `https://gitlab.ikp.kit.edu/AAAlvesJr/corsika`.