Gaudi Evolution for Future Challenges

M. Clemencic, B. Hegner, C. Leggett

CERN Organisation
Europeenne pour la
Recherche Nucleaire
CH-1211 Geneve 23
Switzerland

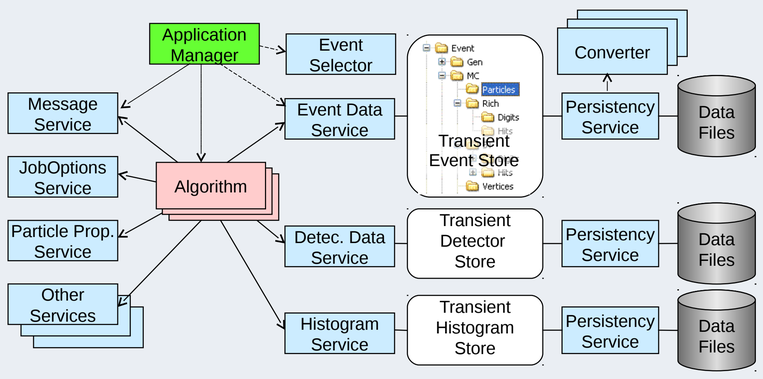
Abstract

The LHCb Software Framework Gaudi was initially designed and developed almost twenty years ago, when computing was very different from today. It has also been used by a variety of other experiments, including ATLAS, Daya Bay, GLAST, HARP, LZ, and MINERVA. Although it has been always actively developed all these years, stability and backward compatibility have been favored, reducing the possibilities of adopting new techniques, like multithreaded processing. R&D efforts like GaudiHive have however shown its potential to cope with the new challenges.
In view of the LHC second Long Shutdown approaching and to prepare for the computing challenges for the Upgrade of the collider and the detectors, now is a perfect moment to review the design of Gaudi and plan future developments of the project. To do this LHCb, ATLAS and the Future Circular Collider communities joined efforts to bring Gaudi forward and prepare it for the upcoming needs of the experiments.
We present here how Gaudi will evolve in the next years and the long term development plans.

Background

Original architecture based on

- Separation between data and algorithms
- Well defined interfaces
- Reusable components



Design principles still valid, but

- Old style C++
- Legacy code
- Not thread safe

Multithreading

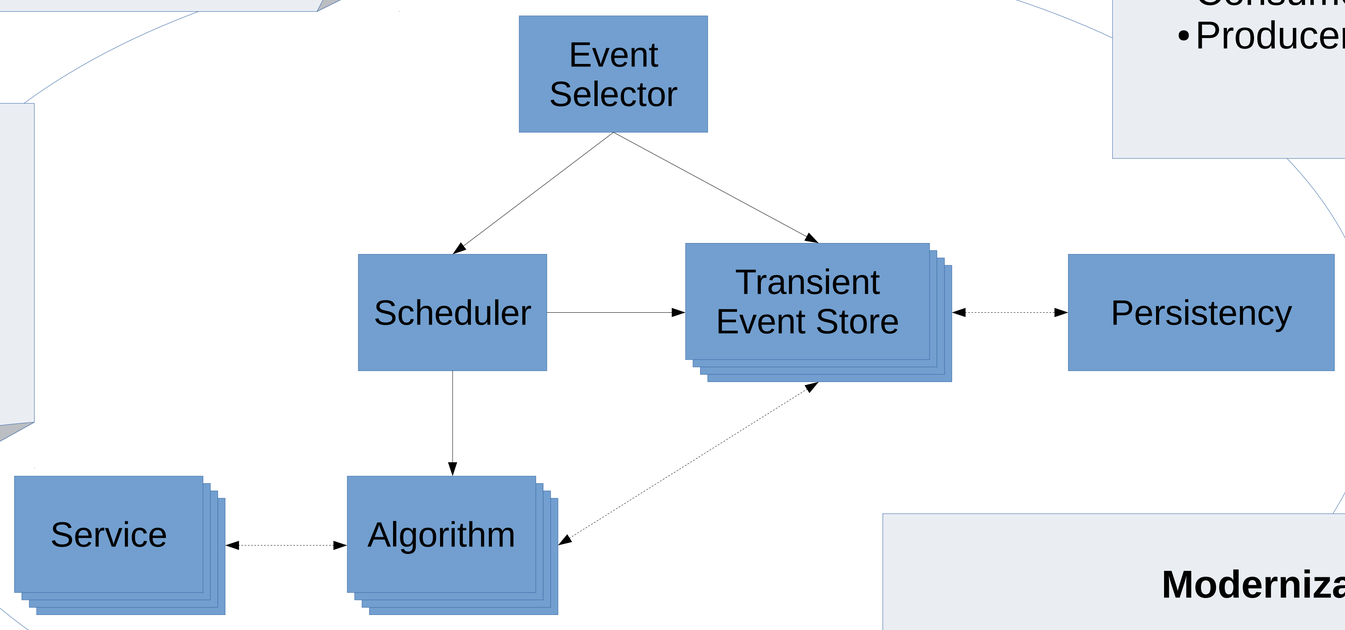
- Task-based parallelism
- Inter- and intra-event parallelism
- Optimized scheduler

Gaudi::Functional

- User defined algorithms as pure functions
- Models for
  - Transformers
  - Filters
  - Consumers
  - Producers

Reentrant Algorithms

- Simplify writing thread safe code
- Communication with Transient Store via “handles”



Configuration

- Review and simplify Python based job configuration
- Simple declaration of control flow constraints

Modernization

- Review use cases
  - Drop code for obsolete use cases
- Improve framework usability
  - Leverage on compiler generated code
- Adoption of C++ Core Guidelines
  - Manage resources by handles
  - …

Gaudi Workshop

Gaudi Workshop 2016

A workshop was held end of September to gather experts from different communities, and re-discuss use cases, best practices, wishes and the design of Gaudi.
What you see in this poster is a short and partial summary of what has been discussed and agreed on at the workshop, where we also defined several action items for developers to pick up.
For more details, please, visit the Gaudi Workshop web page or join us at the Gaudi Developers' Meetings.

Gaudi Developers’ Meetings