ROOT in LHCb Software Framework Gaudi, present and future

M. Clemencic *on behalf of LHCb Collaboration*

September 12, 2018

CERN - LHCb

slides

Table of contents

Use Cases

ROOT, what for?

ROOT is used in Gaudi/LHCb for:

- Persistency and I/O
  - Event Model serialization
  - Analysis exchange format (n-tuples and histograms)
- Analysis
  - Histograms
  - Fitting
- Utilities
  - Matrix functions
  - Vectors
  - Python bindings

ROOT, what NOT for?

We do **NOT** use ROOT as a framework

We use ROOT as a **toolkit**

ROOT in Gaudi so far

The Framework Developer Point of View

- Gaudi is a framework for event driven data processing
- Facilities for creating and filling histograms and n-tuples
- Not really the place to fit data or analyze n-tuples

I'm not going to talk about physics analysis (see previous talks)

I focus on framework related aspects.

Utilities: Linear Algebra

- We use ROOT for matrix and vector operations
  - inversions
  - transformations
  - ...
- Not much to say 🙂
  - math libraries have a modern design and are easy to use
  - at most we *hide* ROOT classes behind aliases

Utilities: Python Bindings

- `cppyy` is great to use C++ from Python
  - interactive exploration of event data
  - prototyping of algorithms and analysis
- We used PyCintex for application bootstrap... before ROOT 6

Utilities: Python Bindings

- `cppyy` is great to use C++ from Python
  - interactive exploration of event data
  - prototyping of algorithms and analysis
- We used PyCintex for application bootstrap... before ROOT 6
- With ROOT 6 the weight of Cling AST is too much
  - we developed an alternative
  - waiting for C++ modules
  - probably will stay with the current alternative

Utilities: Python Bindings

- Generate C++ objects from configuration strings (functors)
  - we pass Python code as job configuration
  - get the pointer to the constructed C++ object
  - use the C++ object in C++ code

Utilities: Python Bindings

- Generate C++ objects from configuration strings (functors)
  - we pass Python code as job configuration
  - get the pointer to the constructed C++ object
  - use the C++ object in C++ code
- The size of the AST forced us to find an alternative
  - cache precompiled versions of objects used in production

Utilities: Python Bindings

- Generate C++ objects from configuration strings (functors)
  - we pass Python code as job configuration
  - get the pointer to the constructed C++ object
  - use the C++ object in C++ code
- The size of the AST forced us to find an alternative
  - cache precompiled versions of objects used in production
- New incarnation of functors under development
  - use Python to generate C++ code
  - JIT compile the code with Cling
  - **but** generated code is not optimized 🙁

Histograms

Nothing special to do, of course!

Histograms

~~Nothing special to do, of course!~~

WRONG!

- Two services to manage histograms are available in Gaudi
  - both work around the *peculiar* memory management of ROOT
  - one exposes `TH*` family of classes (ATLAS choice)
  - the other uses `AIDA::Histogram` (LHCb choice)

N-Tuples (TTree)

- Similar issues as for histograms
  - extremely easy to make mistakes
- LHCb developed a ~~physicist~~ user-friendly interface
  - easy declaration of branches
  - manage correctly the memory for the branches

To be honest...

The main reason for an abstraction for histograms and n-tuples was to transparently switch between ROOT and HBOOK

Anyway HBOOK support has been dropped ages ago.

Future of ROOT in Gaudi

Preparing for ROOT 7

ROOT 7 interfaces have the potential to make
histogram management services pointless.

- We still need a centralize owner of the output file
- Unification of LHCb and ATLAS interfaces under investigation
- First attempts to use ROOT 7 histogram interfaces
  - no segfault!!! 😎
- Looking forward to n-tuples/trees/forests
  - you might have a look to what we have, for inspiration

Summary

Summary

- Not everything is perfect, but you are going in the right direction
- LHCb is undertaking a deep modernization of the code for RUN 3
  - I want ROOT 7: it's so cool!
- I didn't talk about
  - Event Model I/O: important, but is less visible
  - Build System: waiting for new developments