ROOT usage at Belle II

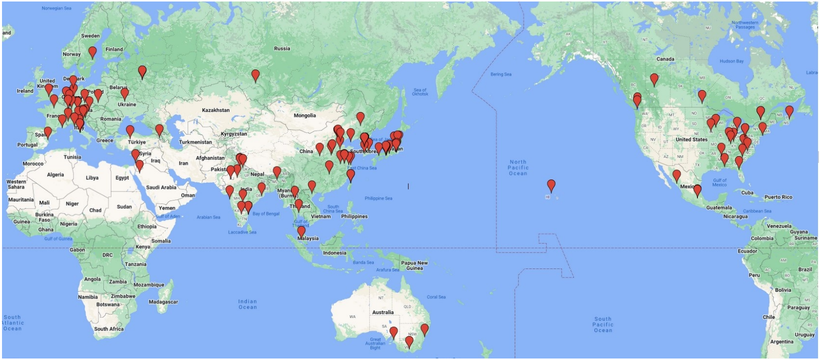
Thomas Lück
on behalf of the Belle II software group

ROOT users workshop 2025

Table of contents

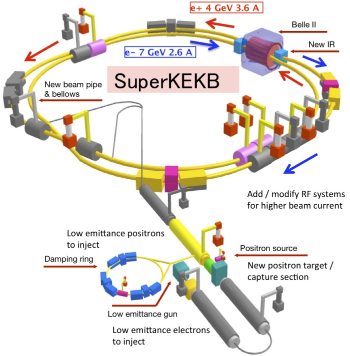
The Belle II collaboration

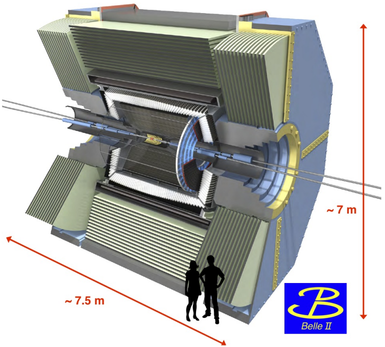
- more than 1000 members
- more than 120 institutions
- 28 countries



The Belle II experiment

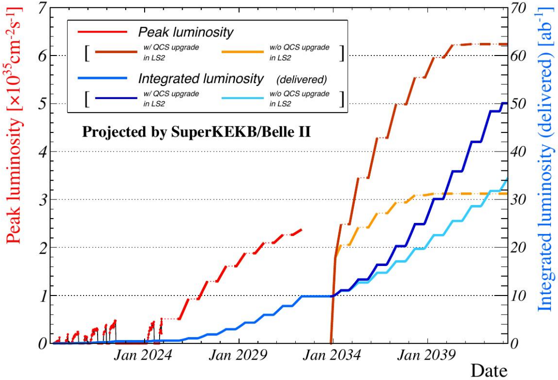
- Belle II next generation B-factory located at KEK (Tsukuba, Japan)
- asymmetric energy e^+e^--collider operating at $\sqrt{s} = 10.58\ GeV$ (mass of $\Upsilon(4S)$)
- rich physics program: B-physics, charm, τ, new physics searches





Belle II dataset

- integrated luminosity up to $575 fb^{-1}$
- target integrated luminosity up to $50 ab^{-1}$
- current world record for instantaneous luminosity $5.1 \times 10^{34} cm^{-2} s^{-1}$
- target instantaneous luminosity: $6 \times 10^{35} cm^{-2} s^{-1}$



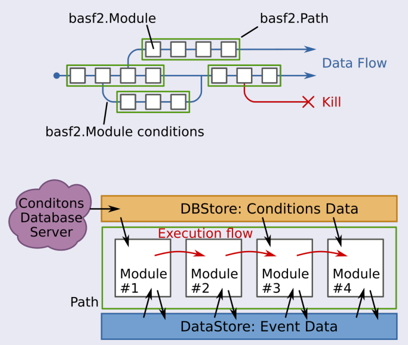
The Belle II analysis software framework (basf2)

- even though called "analysis software" it is not only used for doing analysis
- performs various tasks
  - simulation: event generation and detector simulation
  - reconstruction: tracking, clustering, particle identification
  - analysis: provides analysis tools and predefined variables
  - unpacking of RAW data
- modular software framework written in C++ and links against several external packages:
  - ROOT, Geant4, EvtGen, Pythia, ...
  - various python packages
- basf2 code is open source:
  - code: https://github.com/belle2/basf2
  - documentation: https://software.belle2.org

The Belle II analysis software framework (basf2)

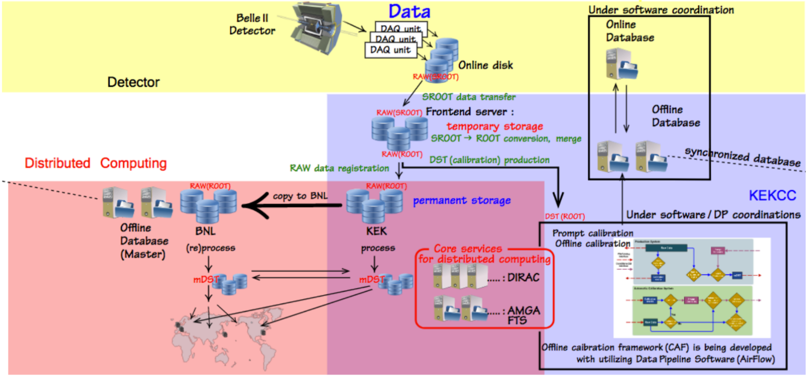
- for steering python is used
- user arranges Modules in a so called path
  - Modules are executed according to position in path
  - paths may fork if needed (e.g. event veto)
- communication between Modules via so called DataStore:
  - lists of objects: StoreArray
  - single objects
- StoreArray:
  - wrapper for ROOT::TClonesArray adding functionality for bookkeeping

Data flow scheme within basf2



Belle II data flow

- data flow until 2023, after 2023 conversion from SROOT to ROOT was removed (next slides)



The RAW data format

- RAW data: directly from the detector recorded at operation

RAW data format before 2023

- custom build sequential ROOT format (SROOT): SROOT files written serialized and uncompressed way
- case of failures: only current event lost, previous events still readable
- ROOT files used for long time storage and data analysis: conversion needed

RAW data format after 2023

- stored data directly in the ROOT format
- file size reduced from 8GB (SROOT) to 3GB (ROOT)
- SROOT lacks in file metadata: using ROOT bookkeeping easier
- data loss only once very beginning of data taking (bug fixed)
- no conversion needed: save computing resources (reduced time till data availability from few hours to 5min)

The Belle II data format

- in general all data is now storted in the ROOT file format
- data files contain two ROOT::TTrees

file metadata:

- TTree with exactly one entry
- Branches contain information on how file was created
  - complete steering file
  - input file name
  - release
  - ...

event data:

- TTree containing event data
- each entry corresponds to one event containing several TBranches
- most information stored as lists (e.g. list of particles, hits, ...): ROOT::TClonesArray
- some event level information using simple types

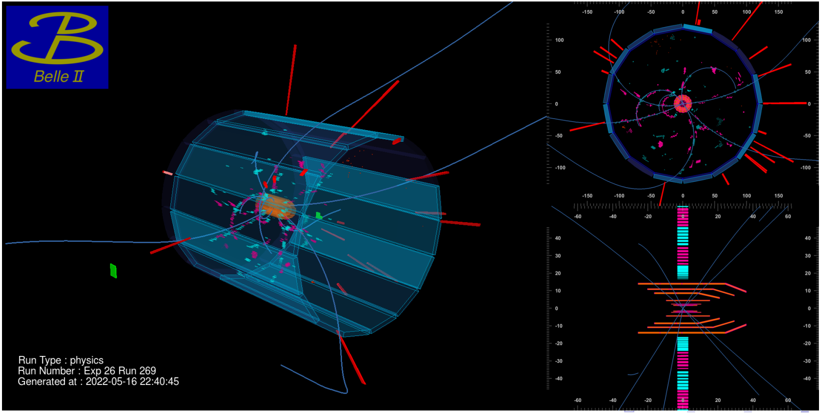
Belle II data formats

- different Belle II data formats differ by which objects stored
- **RAW**: raw data containing the detector information
  - around 70kB/event
  - a total of around 3PB stored in ROOT format up to now
- **cDST**: calibration Data Summary Table
  - around 120 kB/event
  - containing reconstructed objects relevant for calibration
- **mDST**: mini Data Summary Table
  - around 15kB/event
  - containing only data objects needed to do analysis
- **uDST**: user Data Summary Table
  - around 20kB/event
  - mDST objects + additional analysis objects (particle lists)
  - produced during skimming
- use ROOT's schema evolution to provide backwards compatibility

The Belle II conditions database

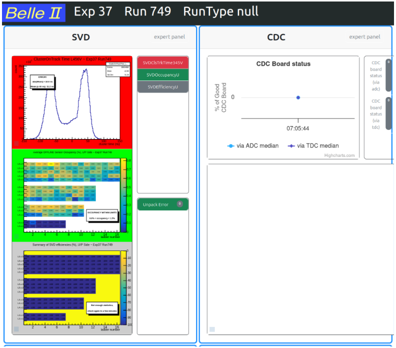
- detector conditions stored in online database (BNL):
  - stores constants and calibrations (e.g. geometry)
  - read at beginning of a processing with basf2
  - mirrored offline in case of connectivity problems
- the software framework assumes that payload contents are serialized ROOT objects
- each database object is its own C++ class inheriting from ROOT::TObject
- automatic generation of ROOT streamers
- stored as ROOT file provided by the database server
- backward compatibility with ROOT's data model evolution scheme

- basf2 has a build in event display
- Geant4 geometry translated to ROOT geometry (using TGeo-classes)
- TEve classes used for visualisation
- online version at https://evdisp.belle2.org



Data Quality Monitoring (DQM)

- Belle II has High Level Trigger software (HLT) running (almost) full reconstruction
- DQM also running on HLT during data taking
- root histograms with various quantities filled to be displayed to control room shifter in real time:
    - detector level quantities: hits, currents, ...
    - high level quantities: reconstructed masses, vertex positions, ...
- histograms stored in ROOT files for later reference



Analysis

- no dedicated analysis framework all included in basf2's analysis package
- common for all working groups (WG)
  - users are encouraged to propagate WG specific code into basf2
  - option to run specialized modules/variables on the grid

What is provided by the analysis package:

- load data
- fill lists of particles created from reconstructed objects
- chose predefined variables to write out
- combine particles to composite particles using string semantics
- perform cut based selections
- apply high level tools:
  - vertex fitting
  - (flavour) tagging
- results are written out as flat candidate based ROOT::TTree

Analysis continued

- once flat ROOT::TTree written by basf2 user is free to use any tool available
- survey showed all kind of types used: C++, ROOT macros, python, jupyter notebooks, ...
- "offline" analyses usually performed at
  - our main computing center (KEKcc)
  - national analysis facilities (DESY, INFN)
  - on local machines
- Disclaimer: My very biased view from the few analyses I have insight!
  - analysts (in particular younger) favor python
  - TTree converted to python dataframes
  - perform pythonian style analysis
  - chose python variants if available: e.g. pyhf instead of RooStats::HistFactory

Summary and outlook

- ROOT is an essential ingredient in our software infrastructure
- constantly trying to improve performance of basf2
- some ideas for improvement but no concrete plans yet:
  - migrate from TTree to RNTuple
  - investigate usage of GPU to accelerate certain tasks
  - use ARM
  - ...
- development team quite small, so need to focus
- using AI methods for handling increasing bkg will become more important:
  - e.g. use GNNs for track finding at Belle II (https://arxiv.org/abs/2411.13596)