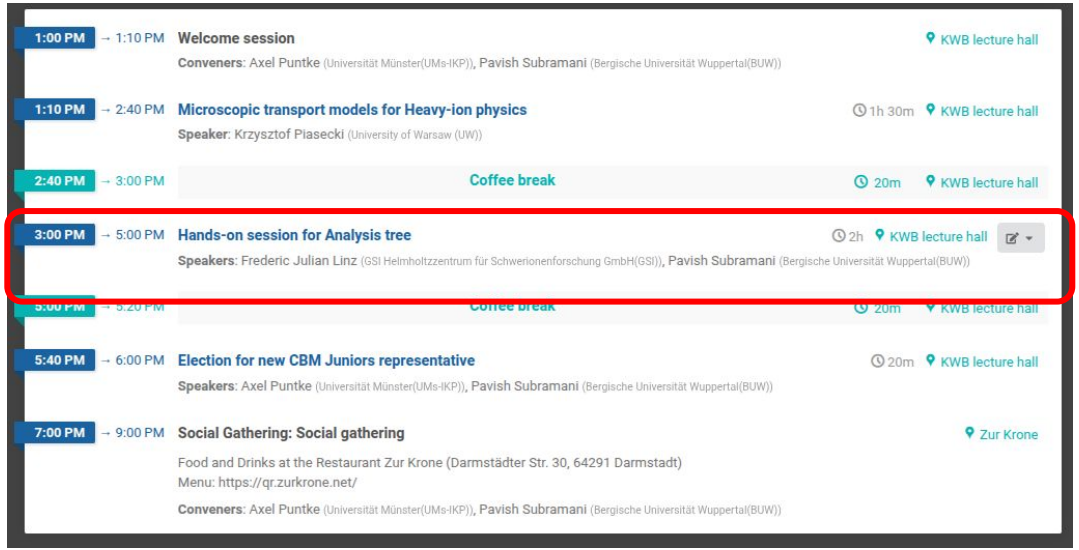
# AnalysisTree - Hands on session

Talk by **Frédéric Julian Linz** (GSI)



| | | | |
|---|---|---|---|
| 1:00 PM | → 1:10 PM | **Welcome session**<br>**Conveners**: Axel Puntke (Universität Münster(UMs-IKP)), Pavish Subramani (Bergische Universität Wuppertal(BUW)) | KWB lecture hall |
| 1:10 PM | → 2:40 PM | **Microscopic transport models for Heavy-ion physics**<br>**Speaker**: Krzysztof Piasecki (University of Warsaw (UW)) | 1h 30m · KWB lecture hall |
| 2:40 PM | → 3:00 PM | Coffee break | 20m · KWB lecture hall |
| 3:00 PM | → 5:00 PM | **Hands-on session for Analysis tree**<br>**Speakers**: Frederic Julian Linz (GSI Helmholtzzentrum für Schwerionenforschung GmbH(GSI)), Pavish Subramani (Bergische Universität Wuppertal(BUW)) | 2h · KWB lecture hall |
| 5:00 PM | → 5:20 PM | Coffee break | 20m · KWB lecture hall |
| 5:40 PM | → 6:00 PM | **Election for new CBM Juniors representative**<br>**Speakers**: Axel Puntke (Universität Münster(UMs-IKP)), Pavish Subramani (Bergische Universität Wuppertal(BUW)) | 20m · KWB lecture hall |
| 7:00 PM | → 9:00 PM | **Social Gathering: Social gathering**<br>Food and Drinks at the Restaurant Zur Krone (Darmstädter Str. 30, 64291 Darmstadt)<br>Menu: https://qr.zurkrone.net/<br>**Conveners**: Axel Puntke (Universität Münster(UMs-IKP)), Pavish Subramani (Bergische Universität Wuppertal(BUW)) | Zur Krone |

CBM CM47 - ECR Day

1st March 2026

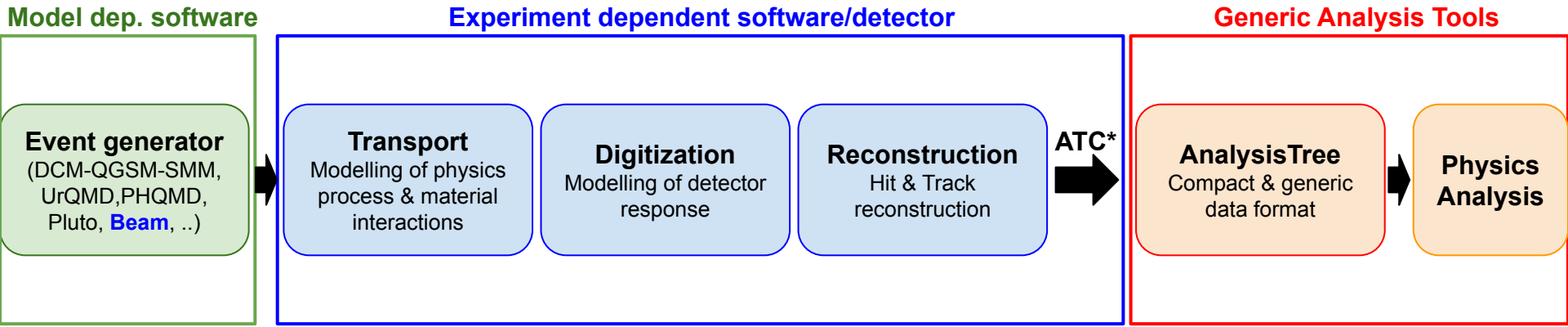
# Outline

- **Introduction**
  - Heavy-Ion Collision Experiment - Data Flow (Simulation) *~3min*
  - AnalysisTree Data Format *~5min*

- **Implementation**
  - AnalysisTree Core Classes *~5min*
  - AnalysisTree Converter (CbmRoot Interface) *~5min*
  - Integration with Analysis Frameworks *~5min*

- **Hands-on session** ***~90min***

# Heavy-Ion Collision Experiment - Data Flow (Simulation)




*AnalysisTree converter

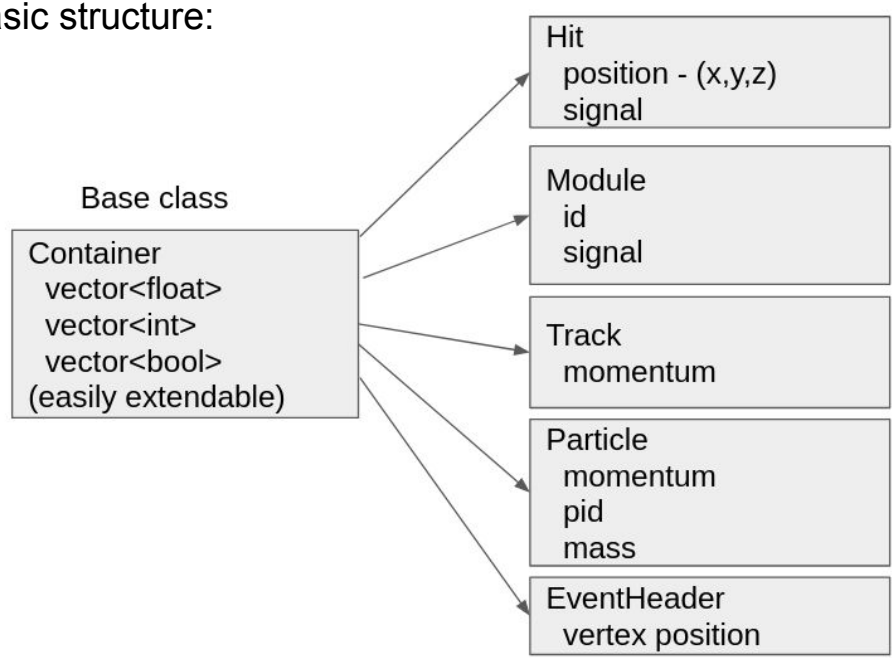
# AnalysisTree Data Format

- Motivation:
  - Compact and generic data format optimized for physics analysis (in heavy-ion collisions)
  - Can be interfaced easily to each detector specific software (different experiments)
  - Common (global) physics analysis tools based on this data format

- Realization:
  - Open source code: https://github.com/HeavyIonAnalysis/AnalysisTree/tree/master

# AnalysisTree Data Format

- Basic structure:




Possible file structure

Detector (template class)
vector<Object>

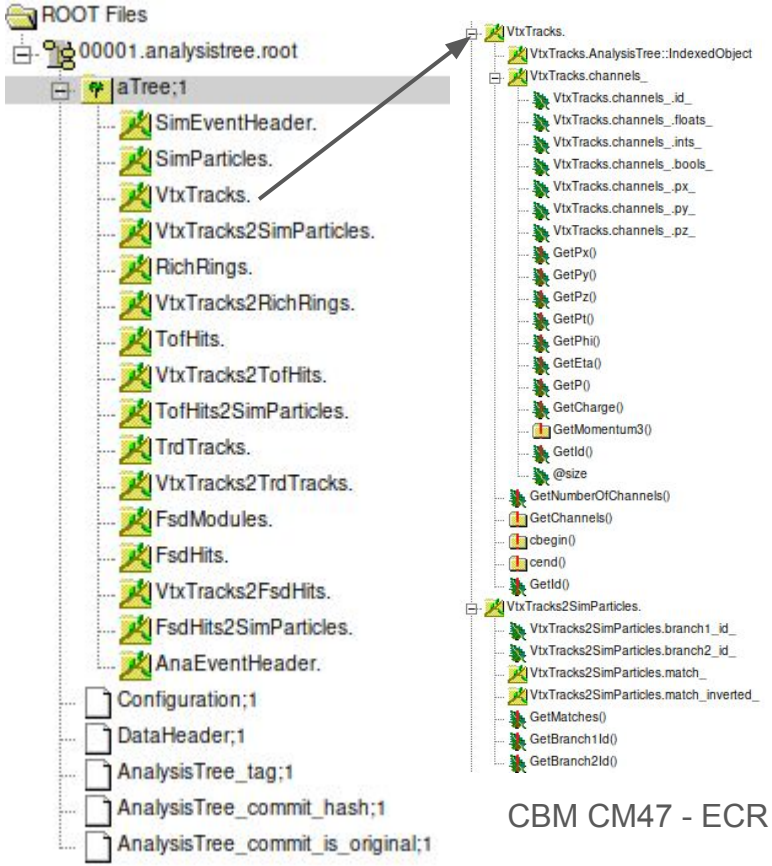
Matching
Detector A <-> Detector B

EventHeader
vertex position, etc

DataHeader
collision system, $p_{beam}$, ...

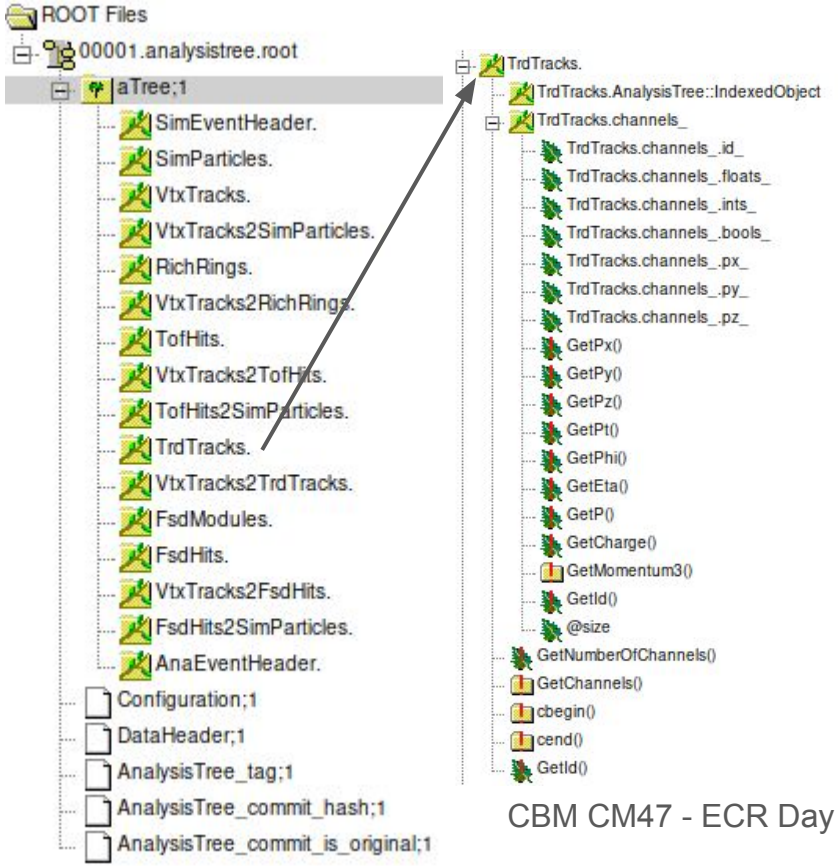
Configuration
Container field names,
List of detectors,
...

# AnalysisTree - CBM File Structure




- Data structure realized in a TTree (ROOT format)
- Individual branches for the detector observables
- Matching between branches in simulation
- DataHeader to store information about run, collision system/energy, etc.
- Filesize reduction for 1000 events:
  - CbmRoot format (tra+digi+reco): ~1.2 GB
  - AnalysisTree format: ~74 MB

# AnalysisTree - CBM File Structure




- Configuration->Print(), e.g. for TrdTracks:

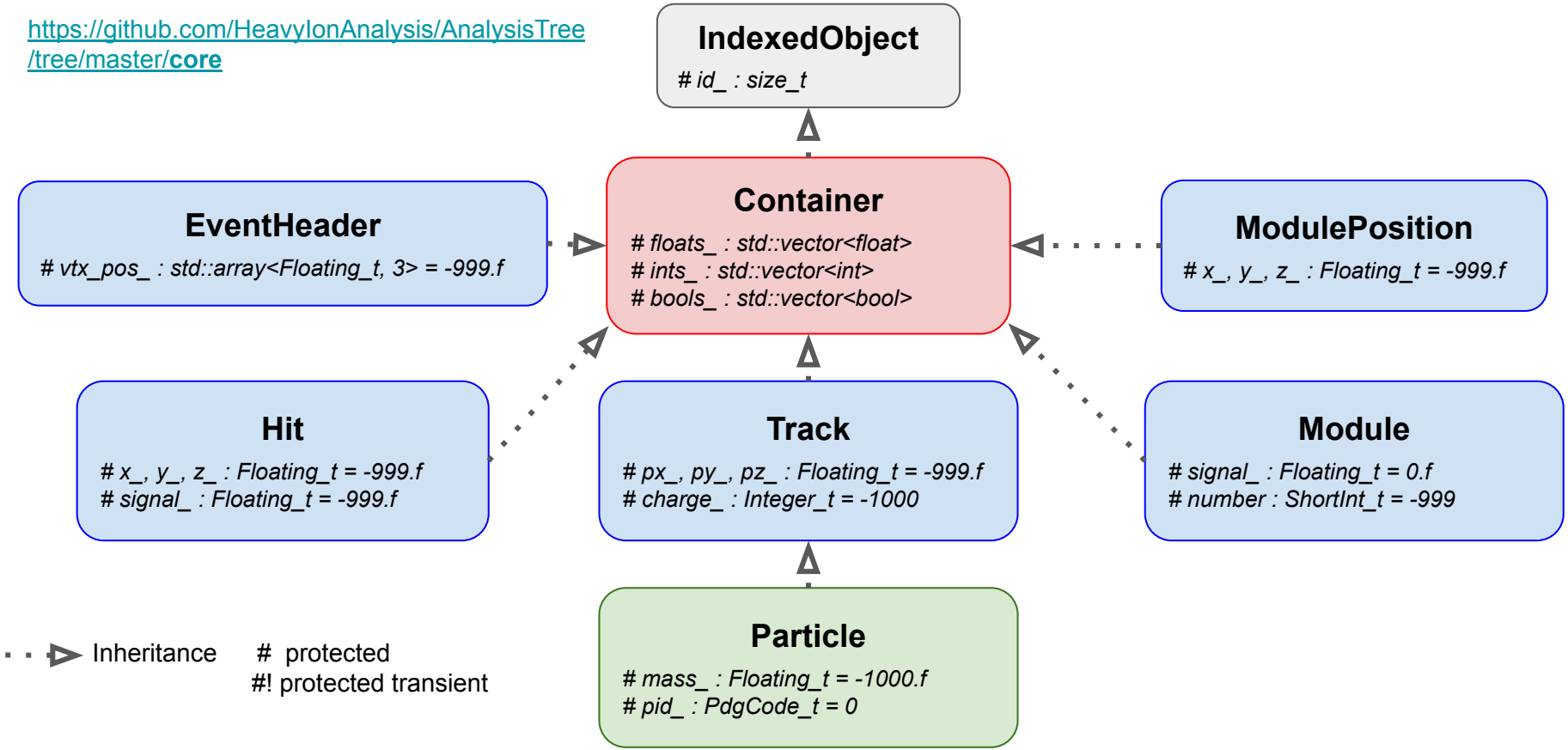
```
Branch TrdTracks () consists of:

Floating fields:
Id        Name             Info
8         chi2_ov_ndf      chi2 divided by NDF of the track fit
0         energy_loss_0    keV(?), Energy loss per TRD station
1         energy_loss_1    keV(?), Energy loss per TRD station
2         energy_loss_2    keV(?), Energy loss per TRD station
3         energy_loss_3    keV(?), Energy loss per TRD station
-3        eta              pseudorapidity
-7        p                GeV/c
-2        pT               GeV/c
9         pT_out           Momentum at last point (?)
10        p_out            Momentum at last point (?)
-1        phi              azimuthal angle
4         pid_like_e       Probability to be a given particle specie
6         pid_like_k       Probability to be a given particle specie
7         pid_like_p       Probability to be a given particle specie
5         pid_like_pi      Probability to be a given particle specie
-4        px               GeV/c
-5        py               GeV/c
-6        pz               GeV/c
```

**Id < 0: AT defined fields; Id ⩾ 0: User-defined fields*

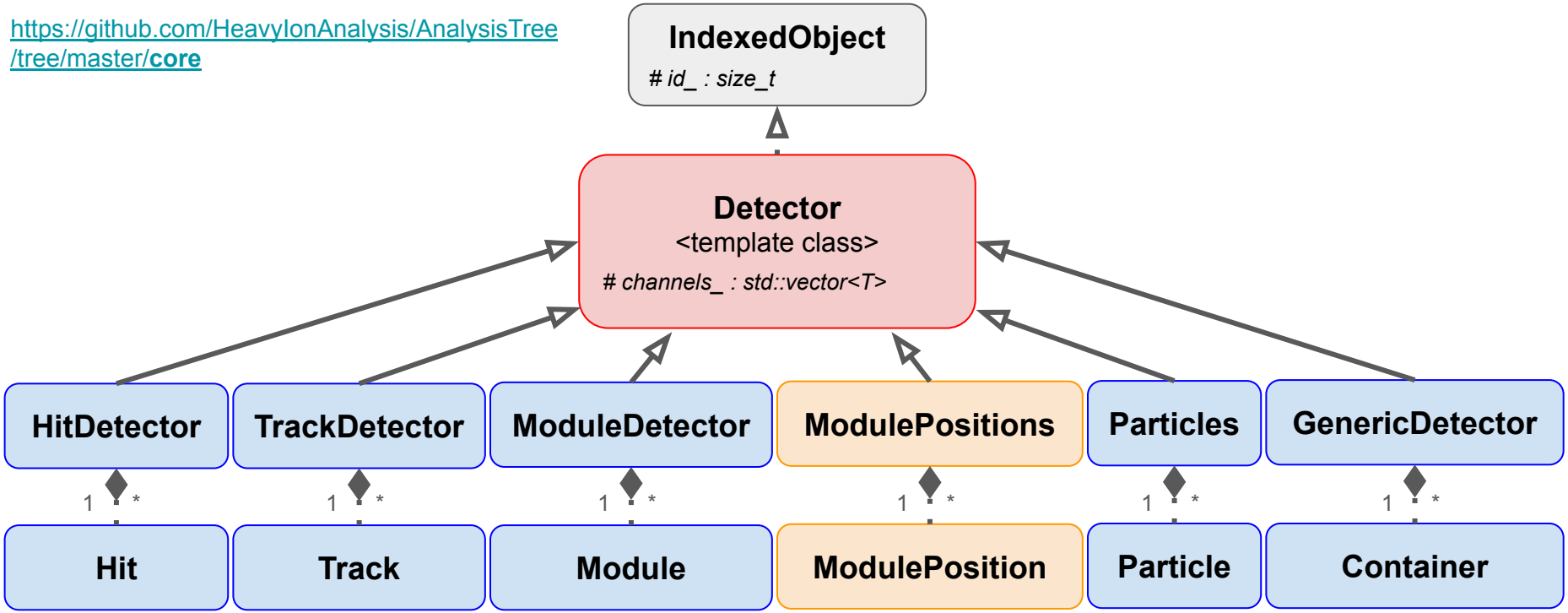
# AnalysisTree - Core *Data* Classes

https://github.com/HeavyIonAnalysis/AnalysisTree/tree/master/**core**

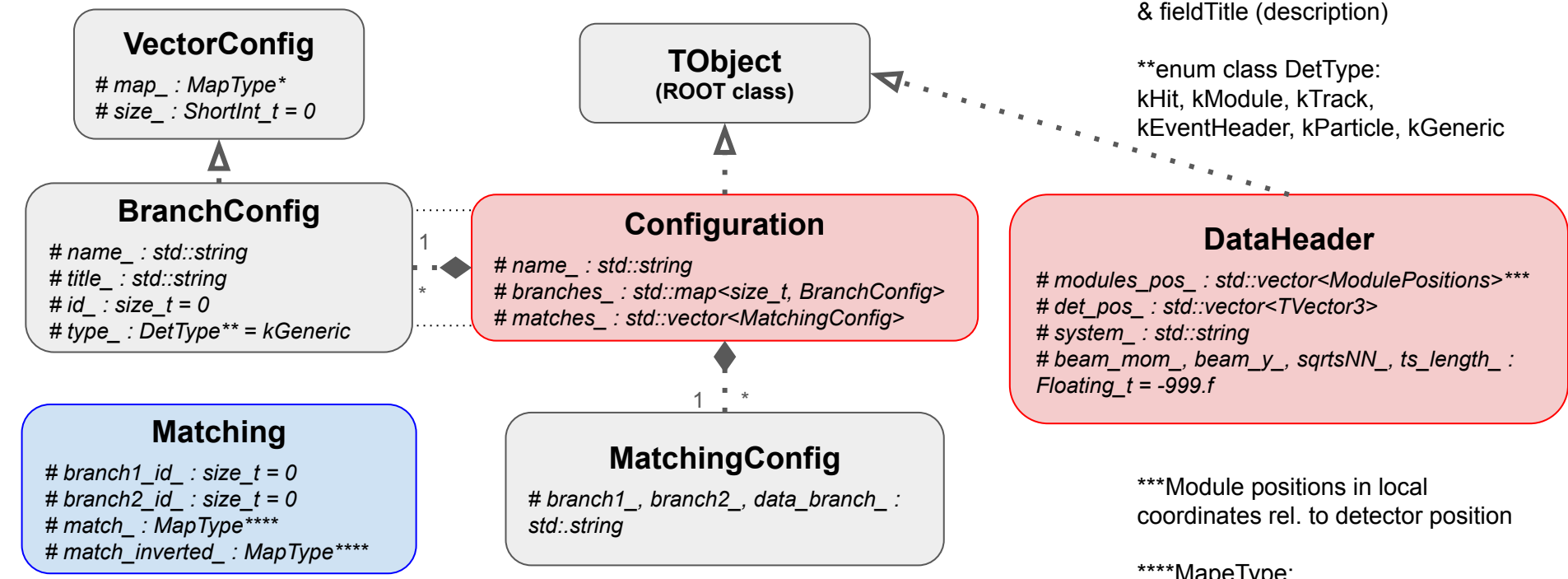


# AnalysisTree - Core *Container* Classes

https://github.com/HeavyIonAnalysis/AnalysisTree/tree/master/**core**



# AnalysisTree - Core *Config* Classes




*MapType:
Mapping between fieldName, fieldId & fieldTitle (description)

**enum class DetType:
kHit, kModule, kTrack, kEventHeader, kParticle, kGeneric

***Module positions in local coordinates rel. to detector position

****MapeType:
Mapping between 2 Integer_t

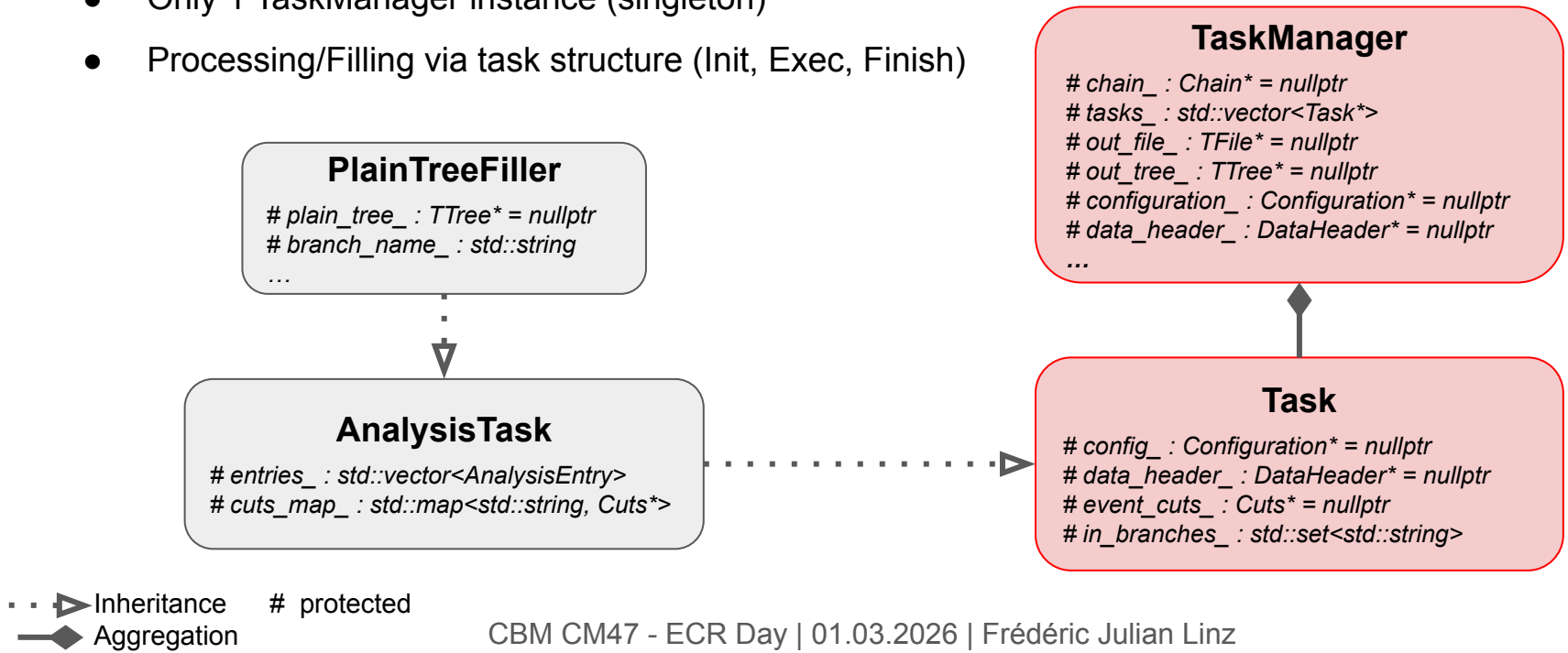
E.g. match of TofHits 14 with SimParticle 2

E.g. match TofHits to SimParticles

Inheritance # protected
Composition

# AnalysisTree - Infrastructure Classes

- TaskManager that manages input data (via Chain), configuration and data header and output
- Only 1 TaskManager instance (singleton)
- Processing/Filling via task structure (Init, Exec, Finish)

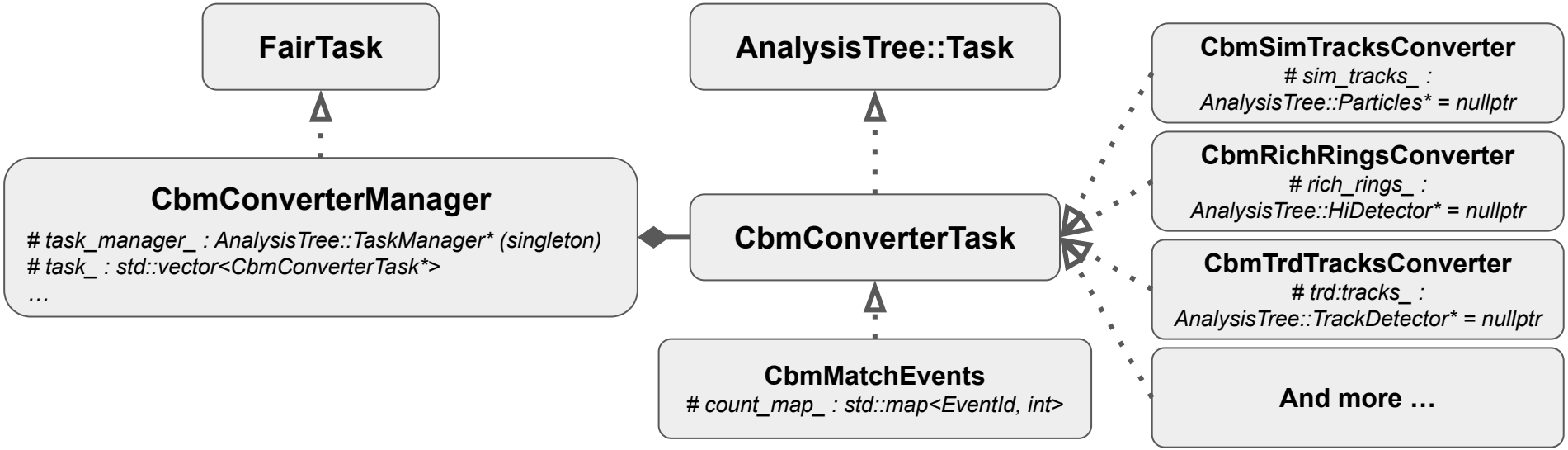


# CbmRoot Interface - AnalysisTree Converter

- AnalysisTree Converter part of CbmRoot: *$CbmRoot_dir/analysis/common/analysis_tree_converter*

  https://git.cbm.gsi.de/computing/cbmroot/-/tree/master/analysis/common/analysis_tree_converter?ref_type=heads

- Task structure used from AnalysisTree::TaskManager, AnalysisTree::Task



# CbmRoot Interface - AnalysisTree Converter

- Individual converter tasks for each detector, e.g. CbmStsTracksConverter
- Init: definition of fields for AnalysisTree output (+id for each field)

```
AnalysisTree::BranchConfig vtx_tracks_config(out_branch_, AnalysisTree::DetType::kTrack);
vtx_tracks_config.AddField<float>("chi2", "spatial chi2 of the track fit");
vtx_tracks_config.AddField<float>("chi2_time", "time chi2 of the track fit");
```

- ProcessData: using (CbmRoot) objects from prior steps (unigen, transport, digitization, reconstruction)
  - Can include post-processing (not just copying data) including possible selection criteria
  - Filling fields by id

```
track.SetMomentum3(momRec);
track.SetField(int(q), iq_);
track.SetField(int(sts_track->GetNDF()), indf_);
```

# CbmRoot Interface - AnalysisTree Converter

- NOTE: CbmMatchEvents - event matching in *time-based mode*:
  - Counting of STS tracks originating from a MC input event:

```
46      count_map_[{file, entry}]++;
```

  - Match events:

```
55    int i {0};
56    for (const auto& match : count_map_) {
57      auto weight = float(match.second) / n_sts_tracks;
58      match_event->AddLink(weight, i++, match.first.entry, match.first.file);
59      LOG(info) << " matched to " << match.first.entry << " " << match.second << " weight = " << weight;
60    }
61
62    if (match_event->GetNofLinks() > 0) event->SetMatch(match_event);
```

  - Event with the most STS tracks will be matched!

# Integration with Analysis Frameworks

- Common directory: https://github.com/HeavyIonAnalysis
- Bayesian particle identification (PID)
  https://github.com/HeavyIonAnalysis/Pid/tree/master/at_interface
  https://github.com/HeavyIonAnalysis/Pid/blob/master/tasks/fill_pid.cpp
- Simple percentile slicing & NBD+MC+Glauber centrality estimation (Centrality)
  https://github.com/HeavyIonAnalysis/Centrality/tree/master/at_interface
  https://github.com/HeavyIonAnalysis/Centrality/blob/master/tasks/fill_centrality.cpp
- Short lived particle decay reconstruction (PFSimple/KFParticleFinder)
  https://github.com/HeavyIonAnalysis/PFSimple/tree/master/at_interface
  https://git.cbm.gsi.de/computing/cbmroot/-/tree/master/analysis/common/at_kfpf_interface
- Flow analysis (QnAnalysis - AnalysisTree interface to QnTools)
  https://github.com/HeavyIonAnalysis/QnAnalysis
  https://github.com/HeavyIonAnalysis/QnTools

# AnalysisTreeQA

- Simple interface to plot 1D/2D histograms & profiles
  https://github.com/HeavyIonAnalysis/AnalysisTreeQA
- Central function in AT-QA macro:
  - Create pointer to TaskManager (singleton)
  - Create pointer to AnalysisTree::QA::Task
  - …
  - Hand this pointer to some function which adds histograms etc. (for readability only)
  - Propagate the task to the TaskManager
  - …
  - Task Management: Init, Run, Finish

```
TaskManager* man = TaskManager::GetInstance();

auto* task = new QA::Task;
task->SetOutputFileName(outfile.data());
const int nEvents = std::stoi(events);

RichRingsQA(*task);
TrdTracksQA(*task);
VertexTracksQA(*task);
TofHitsQA(*task);
SimParticlesQA(*task);
SimEventHeaderQA(*task);
RecEventHeaderQA(*task);
FsdHitsQA(*task);

man->AddTask(task);

man->Init({filelist}, {inputTreeName});
man->Run(nEvents);
man->Finish();
```

# AnalysisTreeQA

- CBM QA for some basic distributions *(needs to be updated)*
  https://git.cbm.gsi.de/computing/cbmroot/-/blob/master/macro/analysis/common/qa/run_analysistree_qa.C

```
task->AddH1({"N_{hits}", {"VtxTracks", "nhits"}, {15, 0, 15}});

// AnalysisTree::Variable in case of more complicated plot
Variable chi2_over_ndf("chi2/ndf", {{"VtxTracks", "chi2"}, {"VtxTracks", "ndf"}},
                       [](  std::vector<double>& var ) { return var.at(0)/var.at(1); });
task->AddH1({"#chi^{2}/NDF", chi2_over_ndf, {100, 0, 100}});

// 2D histo:
task->AddH2({"DCA_{x}", {"VtxTracks", "dcax"}, {100, -1, 1}}, {"DCA_{y}", {"VtxTracks", "dcay"}, {100, -1, 1}});

// Histogramm with additional cut:
Cuts* mc_protons = new Cuts("McProtons", {{"SimTracks","pid"}, 2212});
task->AddH1({"N_{hits}", {"VtxTracks", "nhits"}, {15, 0, 15}}, mc_protons);

// Also Profiles, and more complicated operations
Variable v1("v1", {{"SimParticles", "phi"}, {"SimEventHeader", "psi_RP"}},
            [](std::vector<double> phi) { return cos(phi[0] - phi[1]); });
auto* pid_cut = new Cuts("mc_pion_pos", {{{"SimParticles", "pid"}, 211}});
task->AddProfile({"#it{y}", {particles, "rapidity"}, {20, 0.5, 2.5}}, {"v_{1}", v1, {}}, pid_cut);
```

# Hands-On Session

# Part 1: Setup the Enviroment

- Make sure you have access to /lustre
- Copy the following folder in your lustre directory:

  *cp -r /lustre/cbm/users/fkornas/tutorial /path-to-your-own-dir*

- Set CbmRoot environment (includes AnalysisTree package), e.g. via central installation:

  *source /cvmfs/cbm.gsi.de/debian11/cbmroot/RC/jul25_rc3_111225_v18.8.2_jan24p5/bin/CbmRootConfig.sh*

- Open Root session with some AnalysisTree file, i.e. use the file in the above directory:

  *root -l data/AT/00001.analysistree.root (-l if possible, else use -b instead)*

# Part 2: Get used to AnalysisTree

- Check the configuration and type *Configuration->Print()*
- Check the type of the available branches:

  *Configuration->GetBranchConfig(branch_name).GetType()*
- Draw some fields:

  *aTree->Draw("branch_name.channels_.GetField<type>(id)")**
- Apply some condition (and some option, e.g. same, colz, etc.):

  *aTree->Draw("... ", "branch_name.channels_.GetField<type>(id)==val","option")**
- Draw to histogram:

  *aTree->Draw("...>>hist(nBins,xMin,xMax")*
- Apply some function:

  *aTree->Draw("TMath::Cos(branch_name.channels_.GetPhi())")*

*Use "Scan" instead of "Draw" if you are logged in on virgo

**Drop ".channels_" in case of an event branch

***The pre-defined fields have special function, e.g. GetVertexY(), GetPx(), etc.

# Part 3: Run AnalysisTreeQA

- Open a new terminal (clean environment, access to lustre and SLURM) and go to the copied directory ./tutorial/qa
- Now we will work within the *run_analysistree_qa.C* macro
- For execution (local and on batchfarm) use the bash scripts, i.e. by typing

  *./submit_AnalysisTreeQA.sh config.sh*
- Important: check *config.sh* before (should work out of the box)!

# Part 3: Run AnalysisTreeQA

- First we work on one single file with 1000events (default settings)

  *./submit_AnalysisTreeQA.sh config.sh*

- Modify run_analysistree_qa.C and include (OutputFile1):
  - Include tasks for SimEventHeader and plot basic distributions:
    - Impact parameter, vertex position
  - Include task for RecEventHeader and plot:
    - Chi2_vertex, vertex position, EventHeaderQA
  - 2D Plots:
    - Multiplicity/impact parameter vs. centrality, reco vs. MC vertex position
  - Apply an event selection and remove events where vertex reconstruction failed (OutputFile2)
  - Add centrality cut 0-20% (OutputFile3)

# Part 3: Run AnalysisTreeQA

- Quick comparison between the output files:

  *source /cvmfs/cbm.gsi.de/debian11/cbmroot/RC/jul25_rc3_111225_v18.8.2_jan24p5/bin/CbmRootConfig.sh*

  *cd tools/*
  *mkdir build*
  *cd build/*
  */cvmfs/fairsoft.gsi.de/debian11/fairsoft/jan24p5/bin/cmake*
  *-DBOOST_ROOT=/cvmfs/fairsoft.gsi.de/debian11/fairsoft/jan24p5 -DBoost_NO_SYSTEM_PATHS=ON*
  *-DBoost_NO_BOOST_CMAKE=ON ../*
  *make -j*

  *Clean terminal!*

- Now you should have a binary in the build directory. Available options:

  *./compareRootFiles -h*

# Part 3: Run AnalysisTreeQA

- Now we will implement some particle distributions, therefore set *batch=1* in *config.sh*

  *INPUTDIR=/lustre/cbm/data/sim/mc/urqmd_v4.0/jul25p2_v18.8.2_jan24p5/eb/geant3/auau/pbeam12agev/mbias/day-1/AT/*
- Implement the following histograms (10k events, SLURM):
  - SimParticlesQA:
    - Include SimParticles and plot some basic acceptance plots (AddParticleQA)
    - Add a 2D plot (X,Z) of the particles' starting position (~1bin/cm)
    - Introduce midrapidity and plot pt-y (protons pid=2212, charged pions pid=+-211)
  - VtxTracksQA:
    - Include VtxTracks branch and plots basic properties (AddTrackQA)
    - Add pre-defined matching QA with SimParticles
    - Include pt-y (protons, charged pions)
  - Compare to 0-20% production:

    *INPUTDIR=/lustre/cbm/data/sim/mc/urqmd_v4.0/jul25p2_v18.8.2_jan24p5/auau/pbeam4.41agev/bmax6.71fm/setup2/AT/*