TPC Online Calibration in Run II Status & plans

Ivan Vorobyev

Excellence Cluster Universe

Technische Universität München

ALICE Offline Week

CERN, 19.11.2014

Motivation

❖ TPC in a continuous readout mode in Run 3 will produce a vast amount of data
❖ A data compression factor of ~20 is needed in order to allow for data storage
❖ Online removal of clusters not associated to tracks → <u>online calibration is essential</u>

➢ **<u>Run 2 as R&D phase of online calibration in Run 3</u>**

Conservative scenario:
➢ Port at least CPass0 (mainly TPC calibration) into HLT
➢ Run following passes offline as now using online-created calibration objects

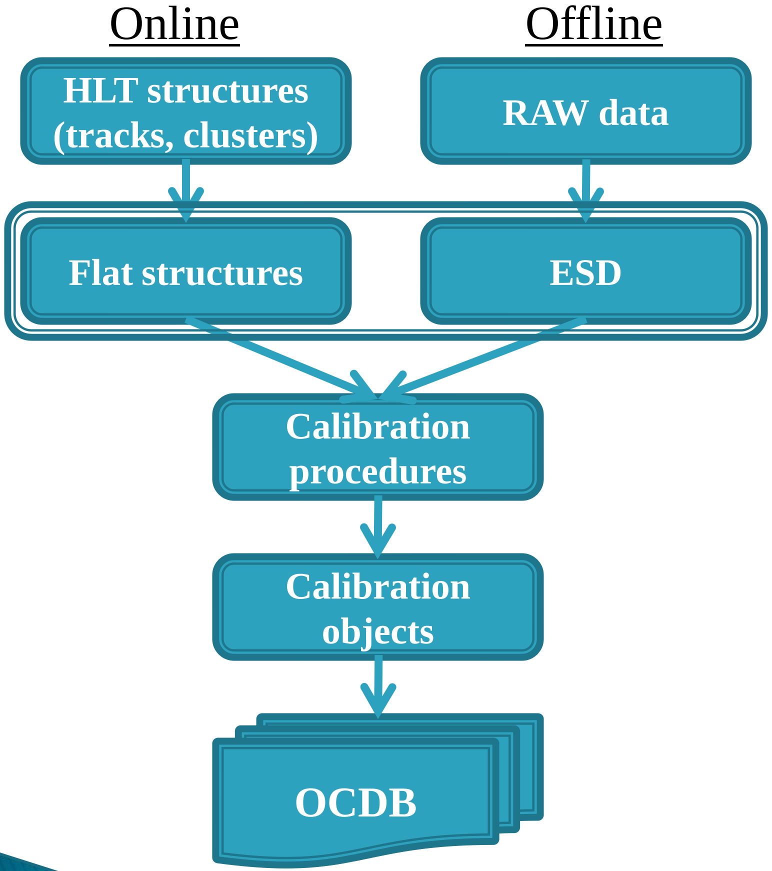
Limitation: ESD object are too complex and heavy for HLT environment

✓ Solution: build special plain (“flat”) objects, similar to HLT structures
✓ Flat objects will be filled in the HLT instead of standard ESD objects

What we need for TPC Calibration:
➢ Common interface between normal ESDs and flatESD objects
➢ HLT component to perform TPC Calibration in HLT environment

Code in flatdev branch

Common interface to work with both FlatESD and standard ESD



❖ Minimum amount of changes in the existing TPC calibration code

❖ Same code running both in HLT and offline

✓ Implementation: AliVEvent/VTrack/Vfriends

✓ TPCCalibTasks use virtual functions in these classes

✓ Getters for v0, Vertex, ExternalTrackParams (next slide)

✓ Tested on local ESD files as input (pp/PbPb)

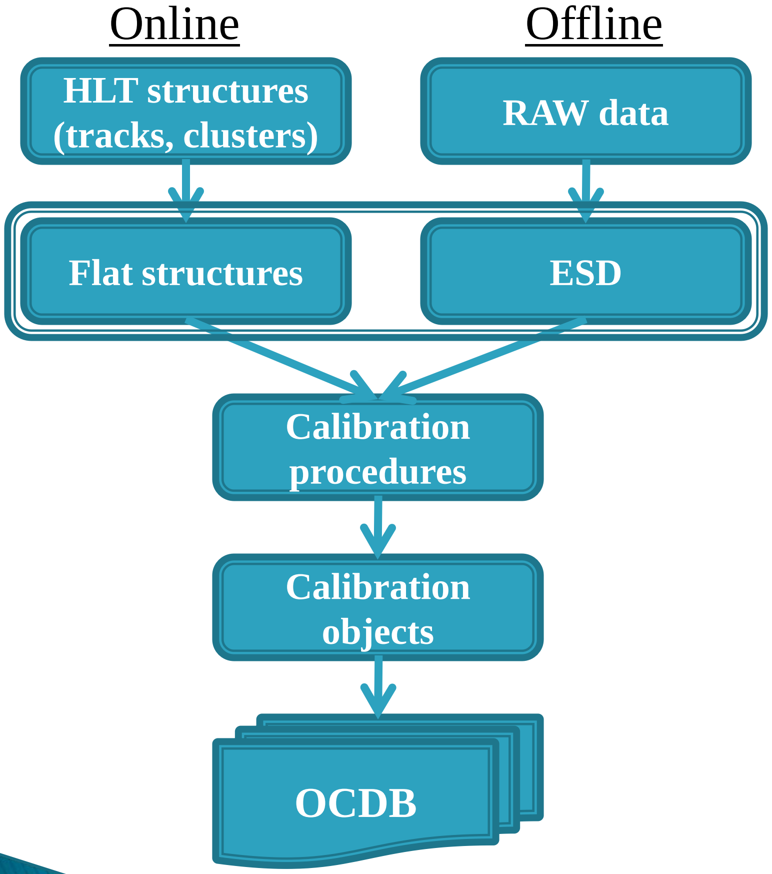
✓ Most of the discrepancies and bugs are fixed, output is the same as for master branch (see next slides)

To do:

➢ More tests with higher statistics in HLT environment

➢ Physics performance, time/memory consumption

Common interface to work with both FlatESD and standard ESD



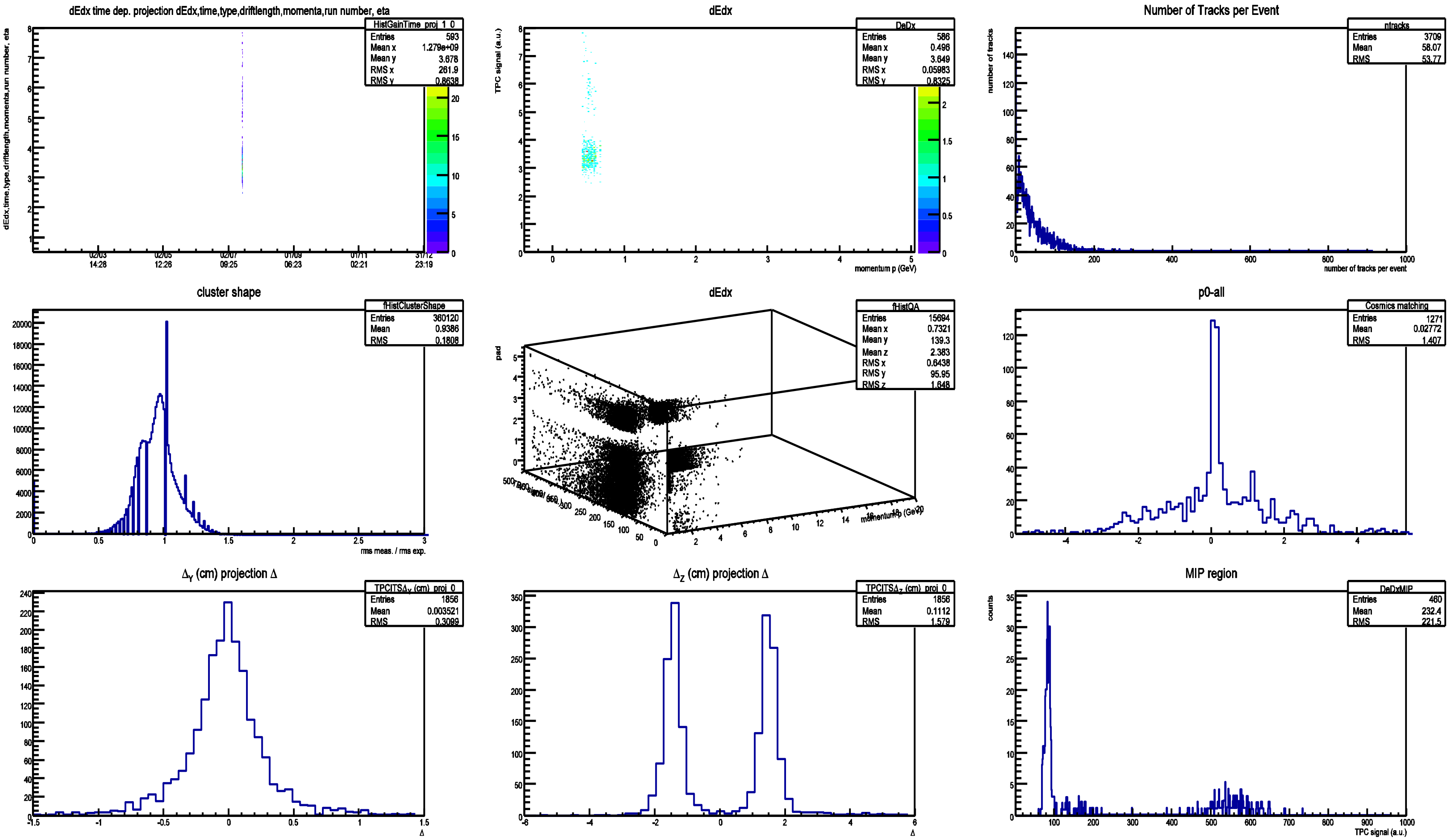
- Special getters for following objects:
  - ✓ AliESDv0
  - ✓ AliESDVertex
  - ✓ AliExternalTrackParam

- One should first instantiate an object on the stack and then fill it with data before use (standard getters returning a pointer will not work in online case with flatESD objects):

```
//AliExternalTrackParam * trackIn;
//trackIn=track→GetInnerParam();
AliExternalTrackParam trckIn;
track → GetTrackParamIp(trckIn);
AliExternalTrackParam * trackIn = &trckIn;

 //(*trackIn)=trackInNew;
track → ResetTrackParamIp(&trackInNew);
```

TPCCluster QA output


dEdx time dep. projection dEdx,time,type,driftlength,momenta,run number, eta

dEdx

Number of Tracks per Event

cluster shape

dEdx

p0-all

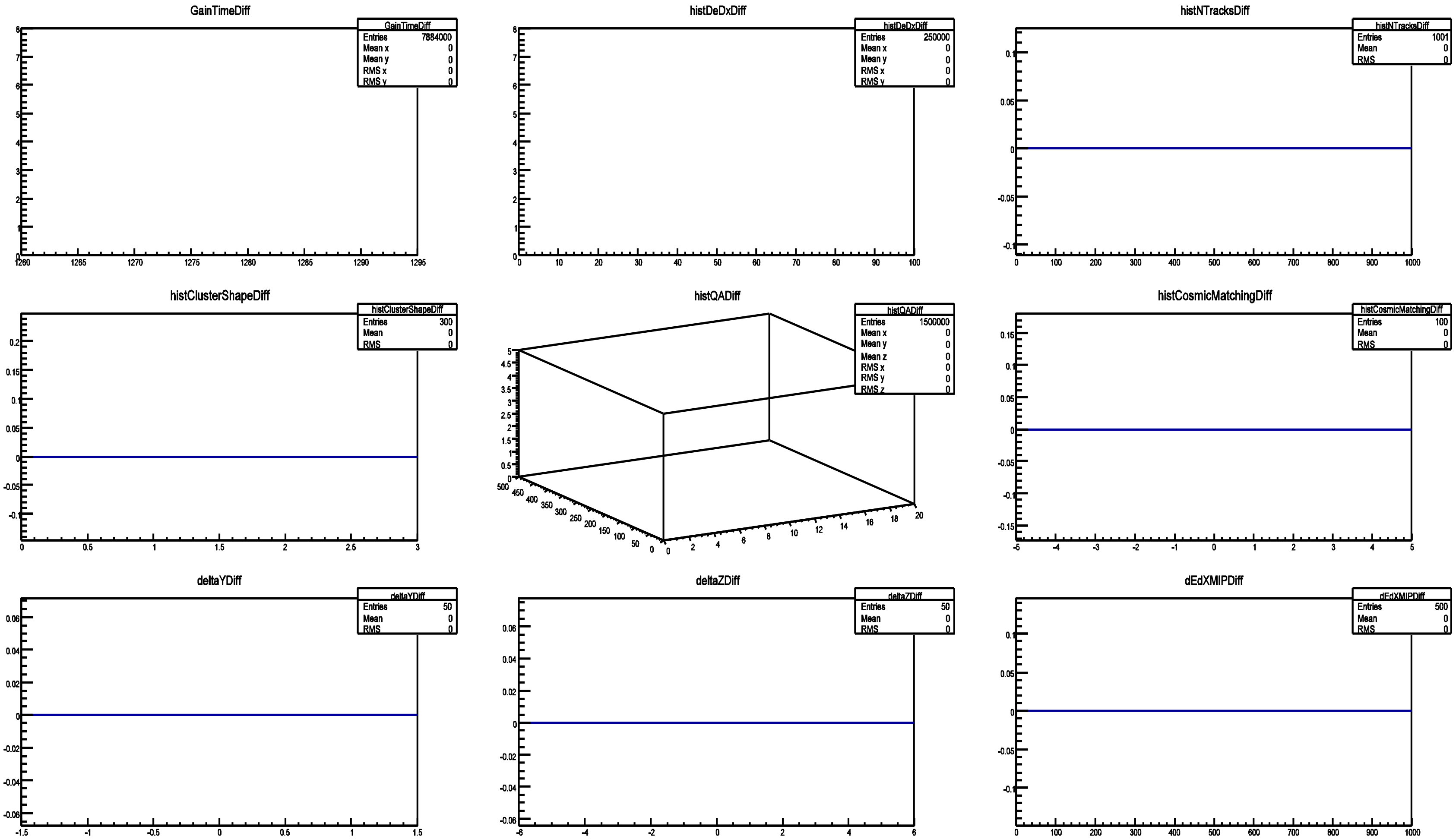
Δ_Y (cm) projection Δ

Δ_Z (cm) projection Δ

MIP region

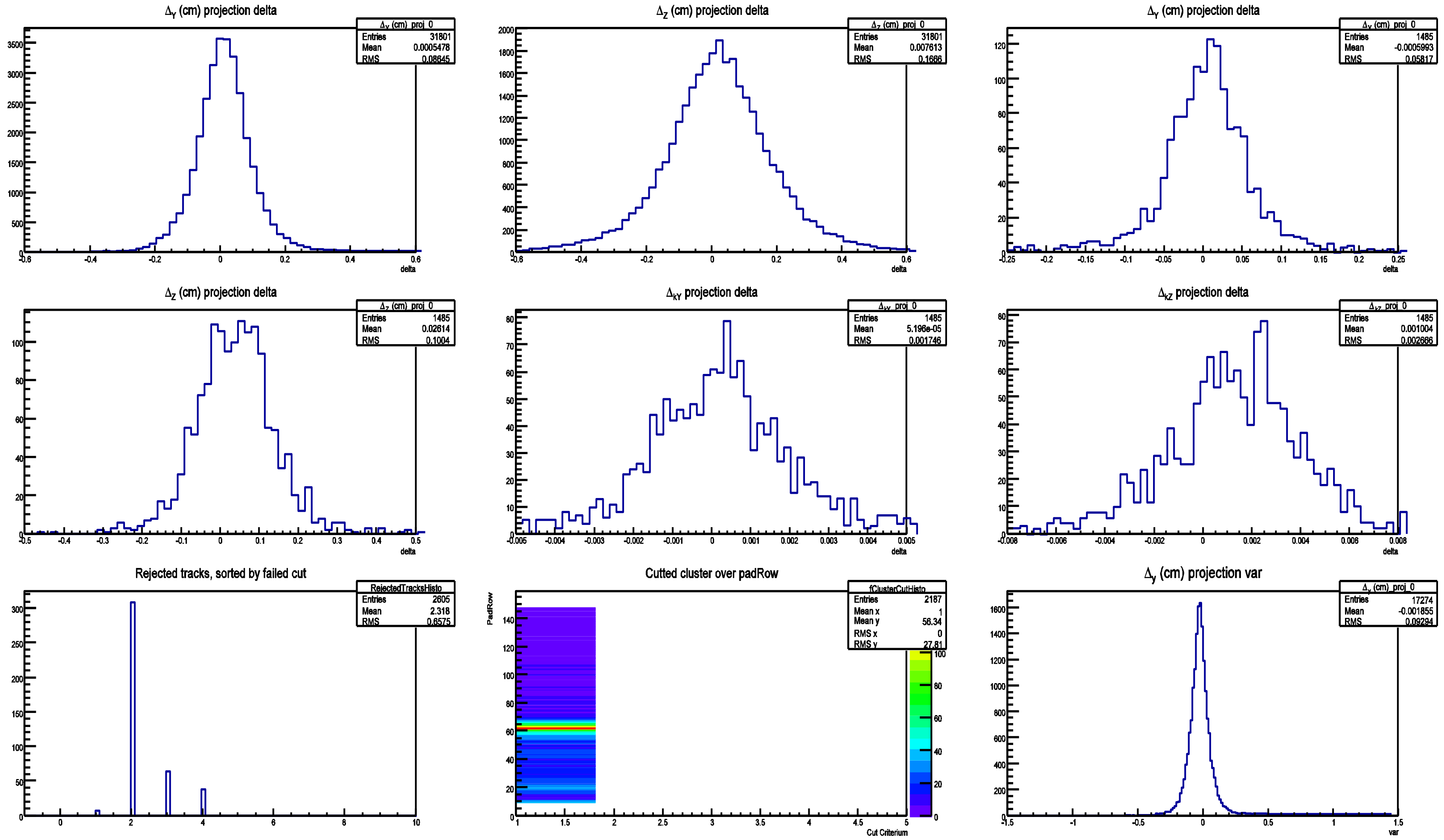
- ❖ Local ESD files as input (pp 7 TeV, 3700 events)
- ❖ Same output as for master branch (next slide)
- ❖ More tests with higher statistics are needed

TPCCluster QA output: difference flatdev − master



- ❖ Local ESD files as input (pp 7 TeV, 3700 events)
- ❖ Same output as for master branch
- ❖ More tests with higher statistics are needed

TPCAlign QA output



- ❖ Local ESD files as input (pp 7 TeV, 3700 events)
- ❖ Same output as for master branch
- ❖ More tests with higher statistics are needed

HLT component for TPC Calibration

❖ Run AliAnalysisManager as in offline case directly in the HLT framework

An example of how to run an analysis manager inside HLT (simple p_T task):
HLT/global/physics/macros/README_AliHLTAnaManagerComponent

Current implementation:
✓ AliHLTTPCCalibManagerComponent: first prototype is ready, at the moment runs one TPCCalibAlign task in HLT environment
✓ Tested on simulated pp data (100 events) on ESD input from GlobalEsdConverter

To do:
➢ Run the tasks on flatESD input (from GlobalFlatEsdConverter) with more statistics
➢ Detailed time/memory consumption: AliSysWatch, valgrind

HLT component for TPC Calibration

❖ Run AliAnalysisManager as in offline case directly in the HLT framework

An example of how to run an analysis manager inside HLT (simple p_T task):
HLT/global/physics/macros/README_AliHLTAnaManagerComponent

Current implementation:

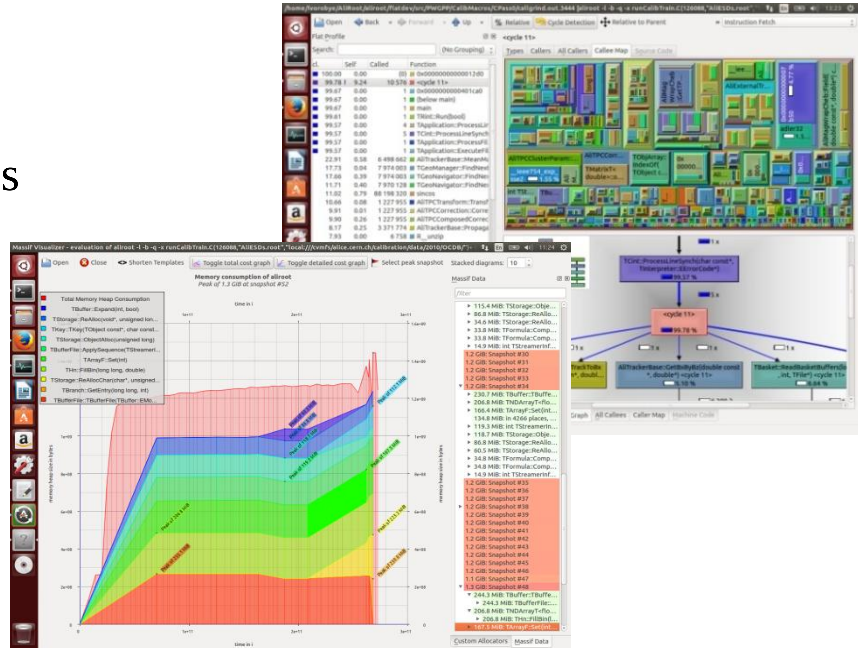
✓ AliHLTTPCCalibManagerComponent: first prototype is ready, at the moment runs one TPCCalibAlign task in HLT environment

✓ Tested on simulated pp data (100 events) on ESD input from GlobalEsdConverter

To do:

➢ Run the tasks on flatESD input (from GlobalFlatEsdConverter) with more statistics

➢ Detailed time/memory consumption: AliSysWatch, valgrind

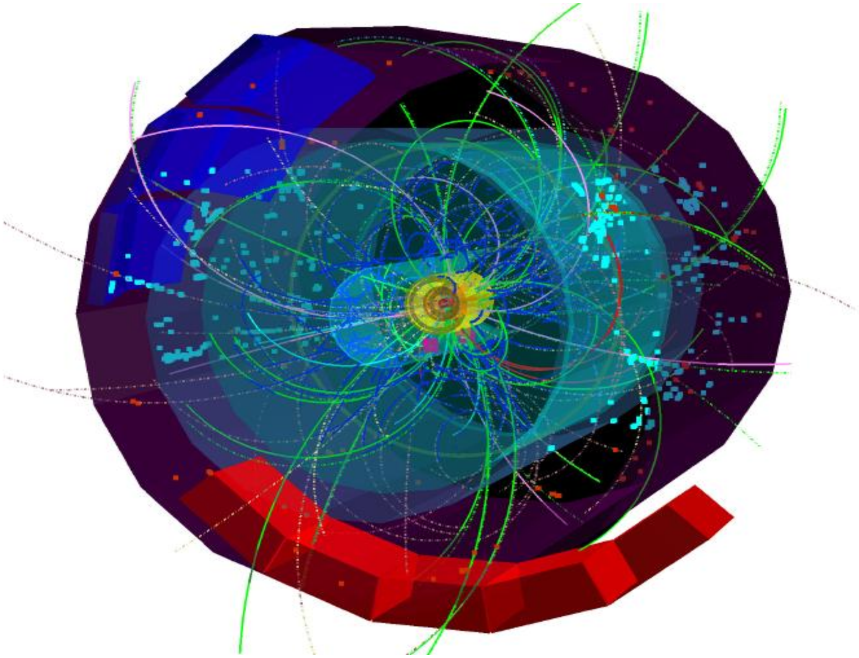

Massif Visualizer - evaluation of aliroot -l -b -q -x runCalibTrain.C(126088,"AliESDs.root","local:///cvmfs/alice.cern.ch/calibration/data/2010/OCDB/")
Open
Close
Shorten Templates
Toggle total cost graph
Toggle detailed cost graph
Select peak snapshot
Stacked diagrams: 10
Memory consumption of aliroot
Peak of 1.3 GiB at snapshot #52
Massif Data
Total Memory Heap Consumption
TBuffer::Expand(int, bool)
TStorage::ReAlloc(void*, unsigned lon...
TKey::TKey(TObject const*, char const...
TStorage::ObjectAlloc(unsigned long)
TBufferFile::ApplySequence(TStreamerI...
TArrayF::Set(int)
THn::FillBin(long long, double)
TStorage::ReAllocChar(char*, unsigned...
TBranch::GetEntry(long long, int)
time in i
Flat Profile
Self
Called
Function
sincos
AliTPCTransform::Transf
AliTPCCorrection::Corre
AliTPCComposedCorrec
AliTrackerBase::Propaga
R__unzip
Types
Callers
All Callers
Callee Map
Source Code
TCint::ProcessLine
<cycle 11>
AliTrackerBase::GetBxByBz(double const*, double*) <cycle 11>
TBasket::ReadBasketBuffers(long long, int, TFile*) <cycle 11>
All Callees
Caller Map
Machine Code

Summary

✓AliVClasses as interface for calibration on Event/Track/Friends level
✓ Special getters for AliESDv0, Vertex, ExternalTrackParam
✓ First prototype of TPC Calibration component in HLT environment
✓ Code in flatdev branch of AliRoot

Next steps:

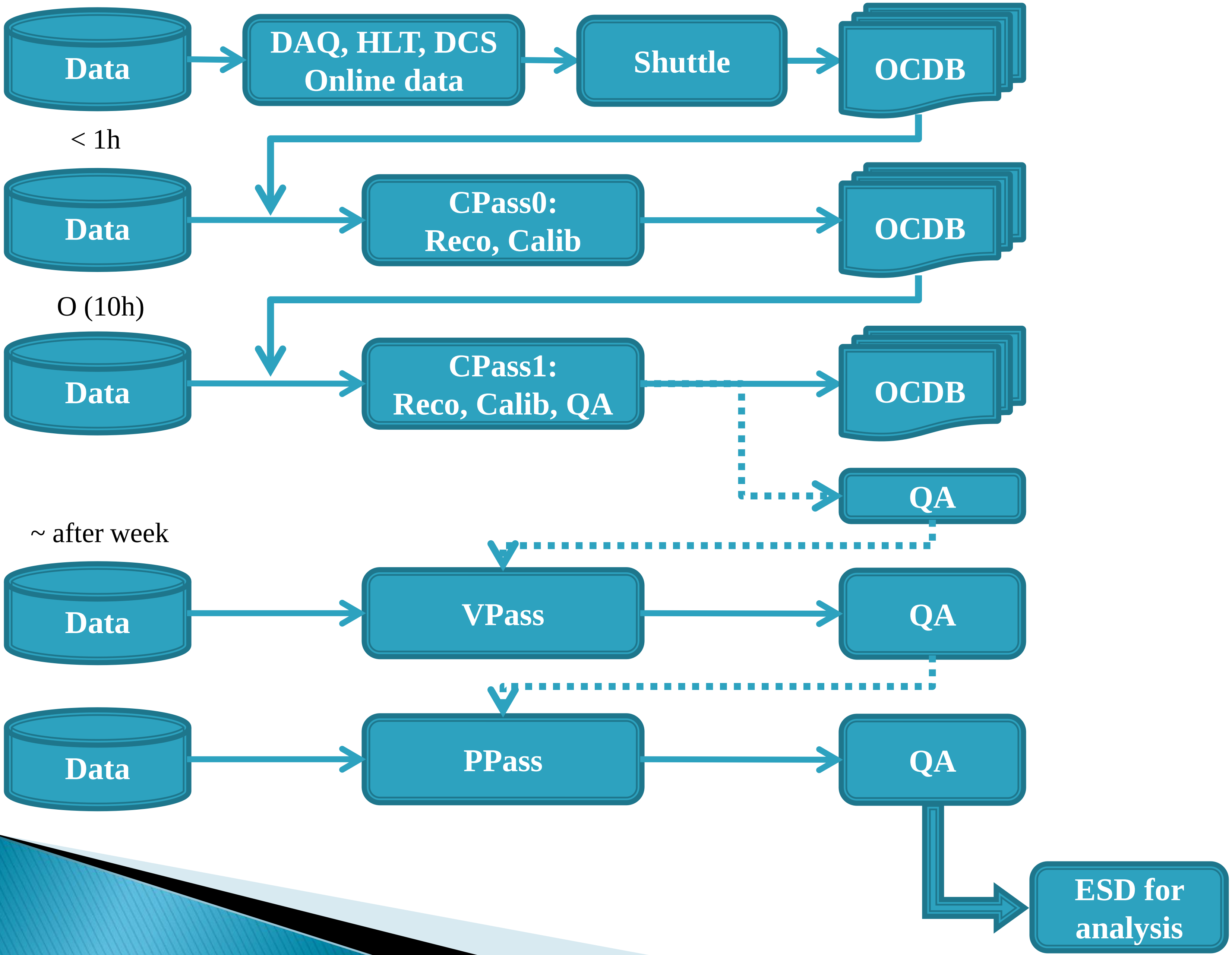
➢ Implement all TPC calibration tasks in HLT component
➢ Tests in HLT environment with more statistics
➢ Run TPC Calibration on flatESD input
➢ Time/memory consumption, bottlenecks
➢ Physics performance



Thank you for attention!

Back-up slides

Run1 Calibration



Data

DAQ, HLT, DCS Online data

Shuttle

OCDB

< 1h

Data

CPass0: Reco, Calib

OCDB

O (10h)

Data

CPass1: Reco, Calib, QA

OCDB

QA

~ after week

Data

VPass

QA

Data

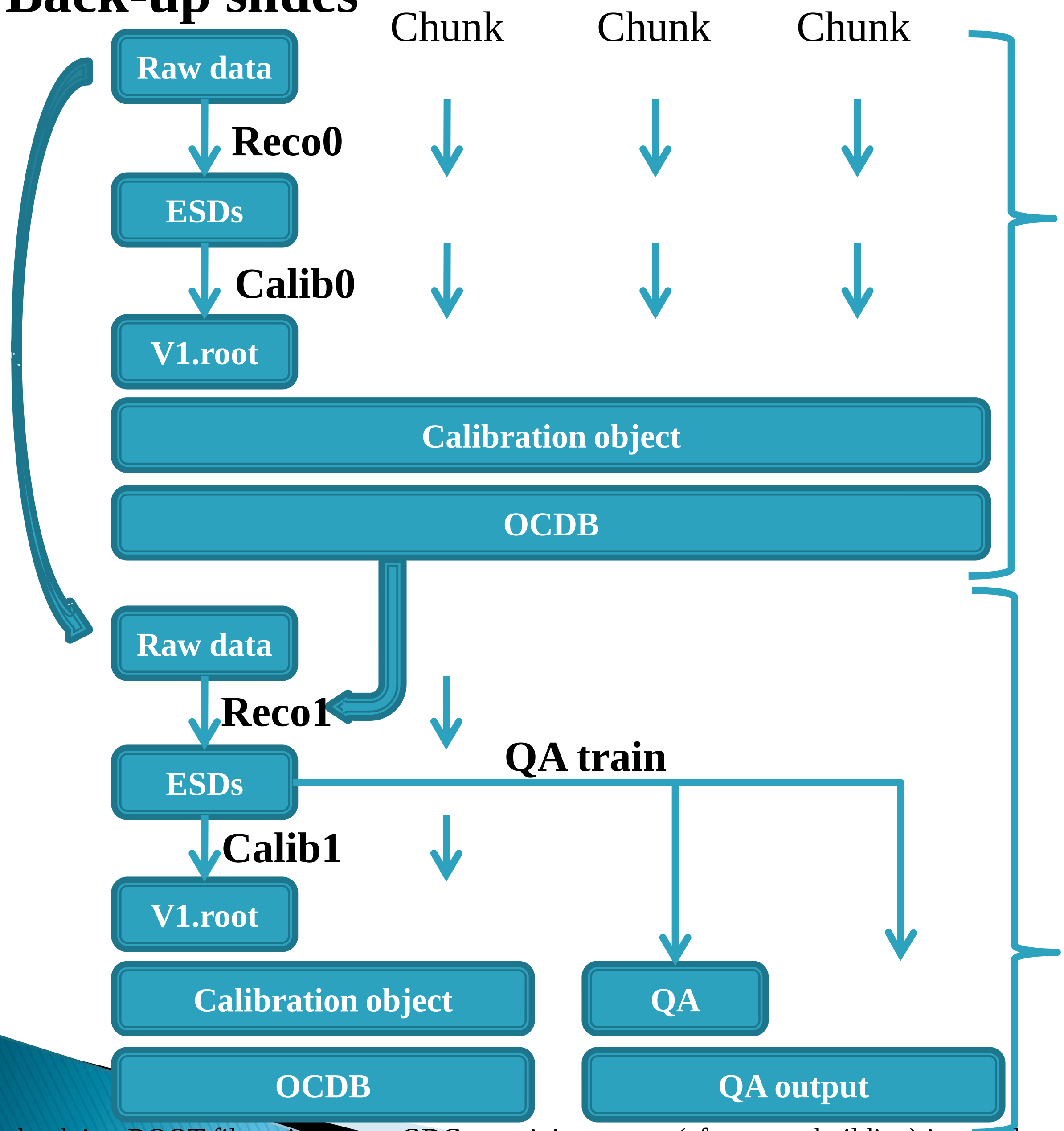
PPass

QA

ESD for analysis

Back-up slides

CPasses in details



Chunk

Chunk

Chunk

Raw data

Reco0

ESDs

Calib0

V1.root

Calibration object

OCDB

Raw data

Reco1

ESDs

QA train

Calib1

V1.root

Calibration object

QA

OCDB

QA output

CPass0

CPass1

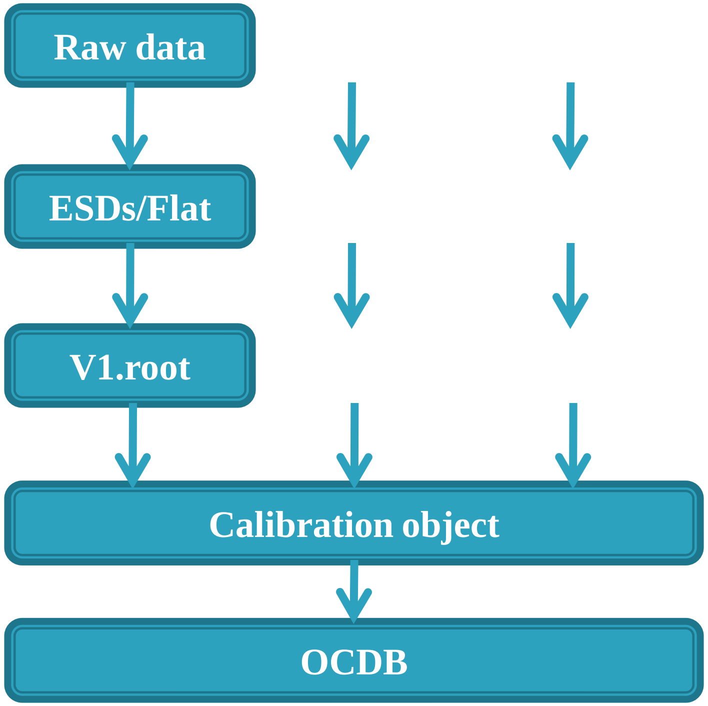
Calibration in ALICE:

- CPass0 – main calibration pass for TPC
- Reco0 and Reco1 ~ same
- Calib0 and Calib1 – differ in configuration only

chunk is a ROOT file written by a GDC containing events (after event building) in raw data format, and stored in the permanent storage (CASTOR) at the end of data-taking. Events are written into different chunks without any temporal order, i. e. , consecutive events may be stored in different chunks. Moreover, there is also no temporal order of events within a chunk.

Back-up slides

Merging of Calibration Objects



❖ HLT Reconstruction & calibration are organized in components, input is a flow of data

❖ Implement a component that receives the data from the calibration ones and merges the output

❖ Merging component will be a "sequential pushing" by the calibration components in some cycles every N-events, not every node at the same time

❖ Time dependent components need to be treated carefully

To do:

➢ Write a merging component and attach it to the task

➢ Reuse Calibration object in offline reconstruction

➢ Run in HLT online environment at LHC P2

Back-up slides

Original idea for interface: VVclasses

• VV interface (for **ALL** involved classes) was invented to avoid TObject inheritance in the flat classes and to provide a uniform interface for offline/online
• reasons to dislike the TObject inheritance of the existing V interface claasses were:
- problematic streaming/reinitializing the TObject
- overhead (8 bytes) for every object

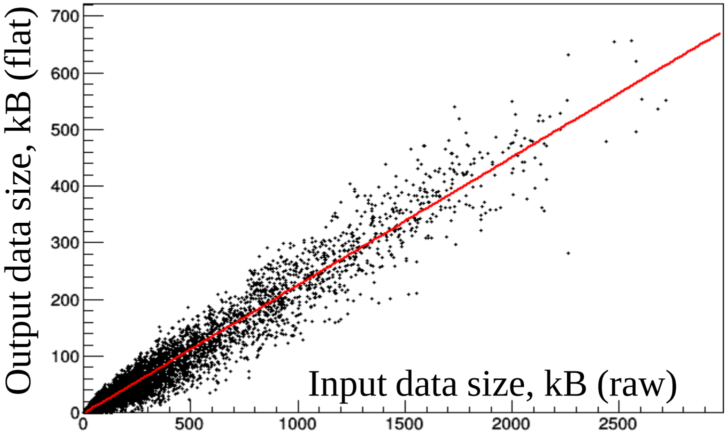
• later we decided to construct externaltrackparams/clusters on the fly to return standard aliroot types which leaves only 4 VV interfaces (Event/Track/friends)
• an efficient TObject streaming method was found eliminating the largest objection to TObject
• problems with keeping 2 virtual sets of interfaces: there are 2 interfaces to maintain + they are similar + double inheritance confuses CINT sometimes

• we get rid of the VV interface altogether and move to standard V classes
• all other contents like external track params, clusters, V0s would be created/filled on the fly using a calibration interface (in practice with little overhead)
• calibration code + QA needs to be ported to use special getters, the standard ones (the ones returning a pointer) will not work in the online

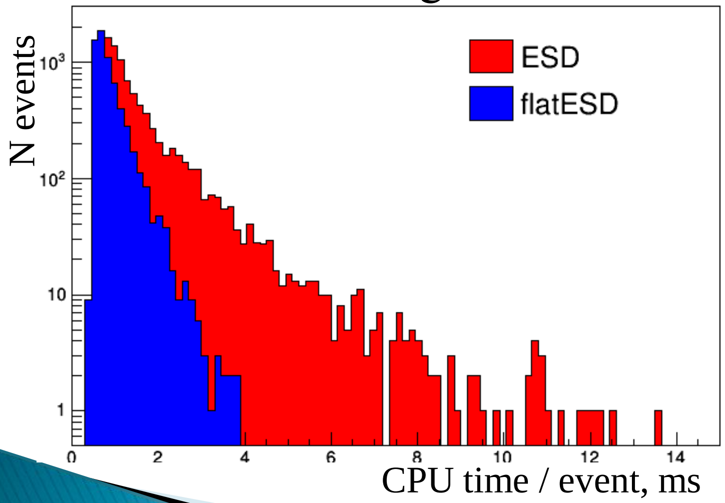
• clean-up of the V interface, to be discussed: make it pure virtual

Creation of flat objects

Tests on pp simulated data (S. Weber):
data size of created flatESDs…



… and CPU timing measurements



❖ create flatESDs from raw data on the fly in HLT to take as input for online calibration
❖ conversion from ESD to flatESD for crosschecks

AliHLTGlobalFlatEsdConverter:
✓ creation of flat ESDs from raw input data
✓ cluster information included
✓ timing measurements show that flatESD creation is significantly faster than ESD creation

AliHLTGlobalEsdConverter:
✓ creation of normal ESDs
✓ cluster information also included

✓ Conversion ESD → flatESD: FlatESDConverter.C

To do:
➢ conversion back from flatESD → ESD
➢ show that ESD → flatESD → ESD conversion chain does not change information

➢ more tests on Pb-Pb events