Utilisation du prototype de ferme d'analyse par Alice

- Le framework d'analyse d'Alice
- Ce qui a été fait sur la LAF
- Ce qui doit être fait sur la LAF

Laurent Aphecetche
Subatech

« Dumb tests »

~~Utilisation~~ du prototype de ferme d'analyse par ~~Alice~~

« 1 user »

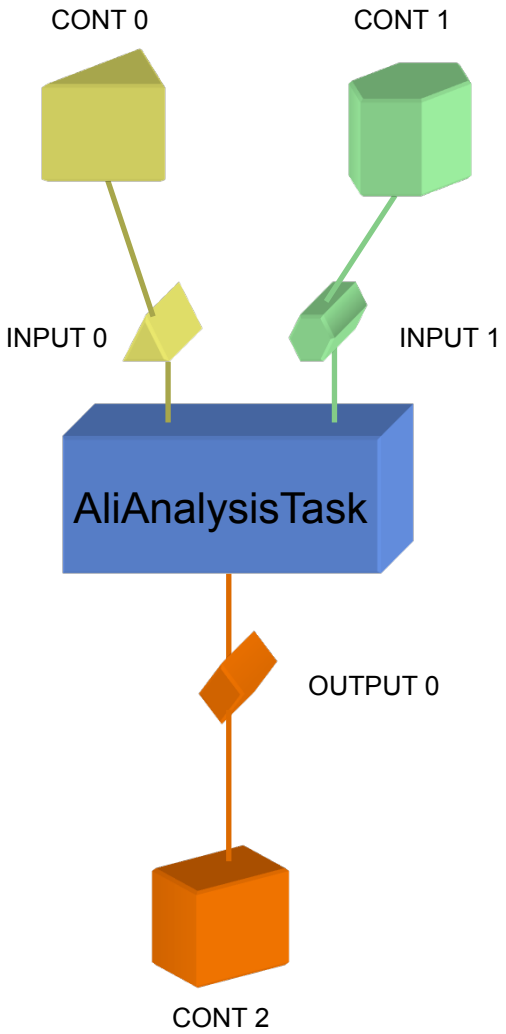
What the Analysis framework does in ALICE

- Transparent access to all resources with the same code
  - Usage: Local, AliEn grid, CAF/PROOF
- Transparent access to different inputs
  - ESD, AOD, Kinematics tree (MC truth)
- Allow for „scheduled" analysis
  - Common and well tested environment to run several tasks
- Defines a common terminology

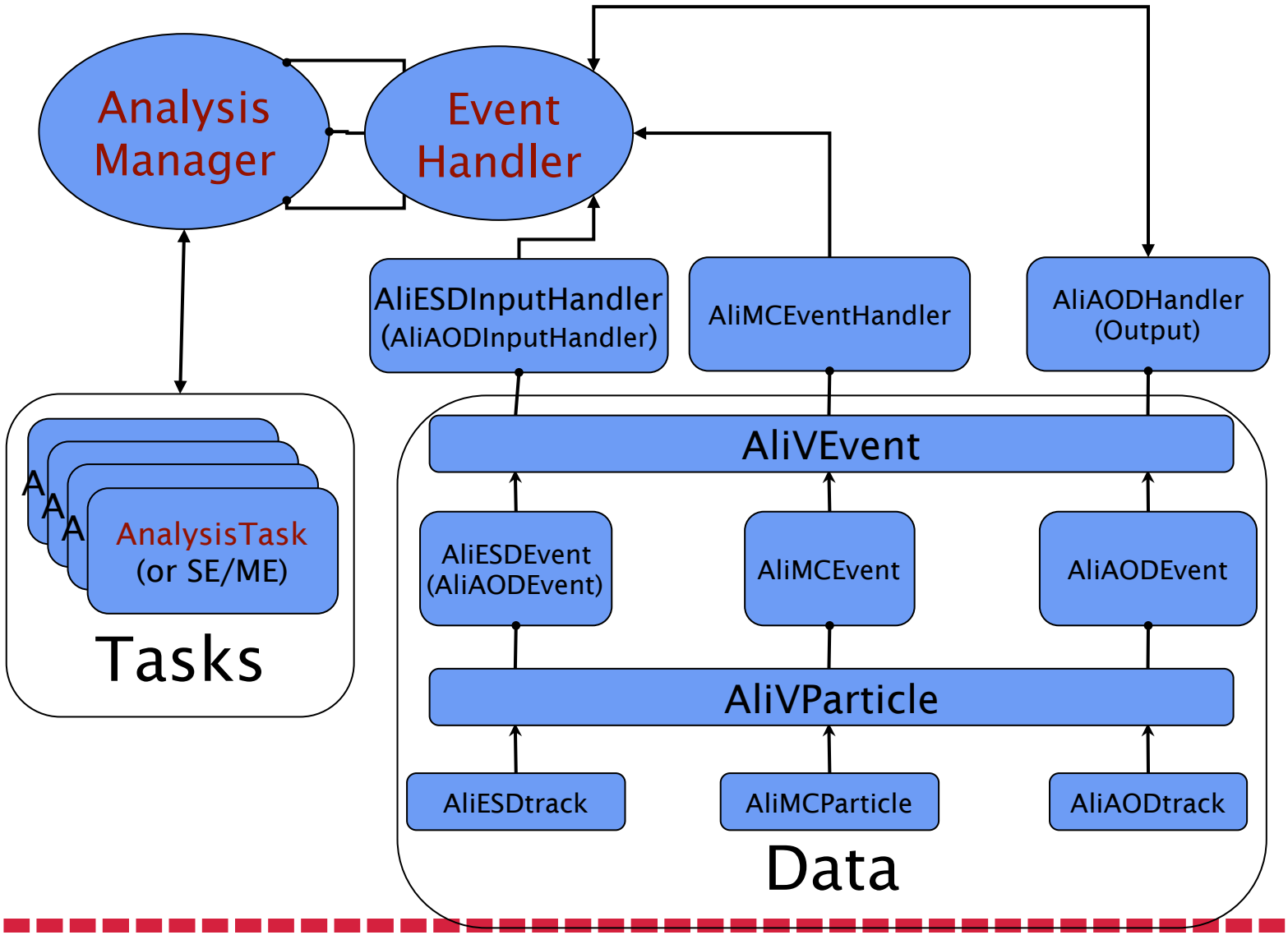
N.B.: The analysis framework itself has a very general design, not bound to ALICE software

The single task view

- AliAnalyisTask
  - User provided code
- Input data
  - Provided via numbered slots
  - Each slot connected to a data container of the corresponding type at run time
  - Content can be any TObject
  - "Handlers" handle data specific operations
- Output data
  - Communicated via one or more slots
  - Handlers e.g. for AOD output
  - Simpler output e.g. histograms
  - Output can be disk resident (file) or only memory resident (transient data)
- **Several of these tasks can be collected in the *manager***



The overall picture

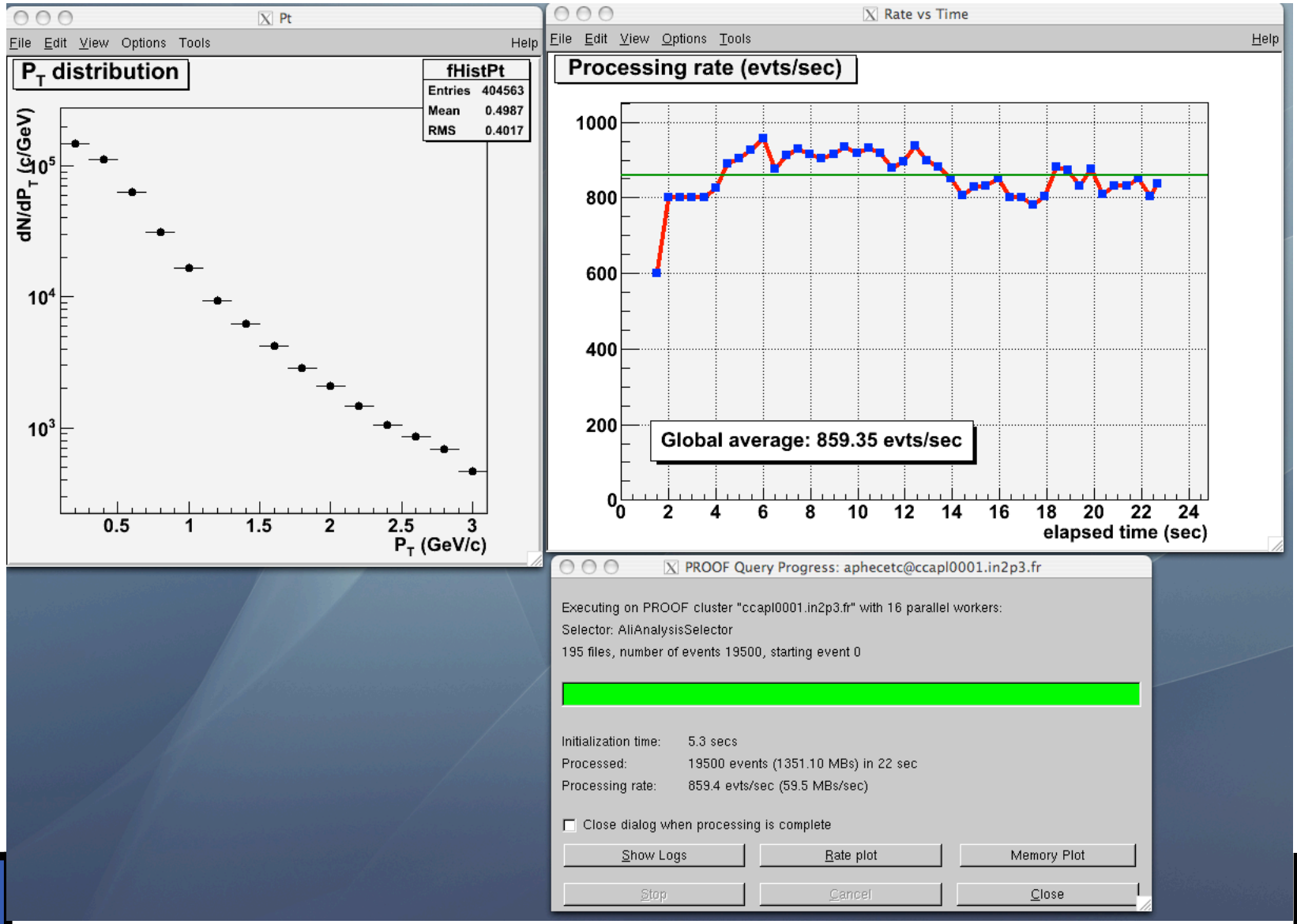


Ce qui a été fait

- Importation de données Alice sur LAF (« à la main ») (195 fichiers ESDs)
- Installation suite logicielle spécifique Alice : alien (client) + root + geant3+ aliroot sur zone afs dédiée
- Une analyse de test de base (boucle sur les données et fait un plot de la distribution en pt des particules…)
- LAF V1 = 16 coeurs

```
{
TProof::Open("ccapl0001");
gProof->GetManager()->SetROOTVersion("v5-23-04");
gProof->UploadPackage("AF-v4-16"); gProof->EnablePackage("AF-v4-16");
mgr = new AliAnalysisManager("testAnalysis"); // Create the analysis
   manager
gProof->Load("AliAnalysisTaskPt.cxx++g");
task = new AliAnalysisTaskPt("taskpt"); // Create, add task
mgr->AddTask(task);
AliESDInputHandler* esdH = new AliESDInputHandler; // Add ESD handler
mgr->SetInputEventHandler(esdH);
gROOT->LoadMacro("CreateESDChain.C");
chain = CreateESDChain("filelist.txt",200);
cInput = mgr->CreateContainer("cInput", TChain::Class(),
   AliAnalysisManager::kInputContainer);
mgr->ConnectInput(task, 0, cInput);
cOutput= mgr->CreateContainer("cOutput", TH1::Class(),
   AliAnalysisManager::kOutputContainer, "Pt.root");
mgr->ConnectOutput(task, 0, cOutput);
mgr->InitAnalysis();
mgr->StartAnalysis("proof", chain);
}
```

« Résultats »


Pt
File Edit View Options Tools Help
P_T distribution
fHistPt
Entries 404563
Mean 0.4987
RMS 0.4017
dN/dP_T (c/GeV)
10^5
10^4
10^3
0.5 1 1.5 2 2.5 3
P_T (GeV/c)
Rate vs Time
File Edit View Options Tools Help
Processing rate (evts/sec)
1000 800 600 400 200 0
0 2 4 6 8 10 12 14 16 18 20 22 24
elapsed time (sec)
Global average: 859.35 evts/sec
PROOF Query Progress: aphecetc@ccapl0001.in2p3.fr
Executing on PROOF cluster "ccapl0001.in2p3.fr" with 16 parallel workers:
Selector: AliAnalysisSelector
195 files, number of events 19500, starting event 0
Initialization time: 5.3 secs
Processed: 19500 events (1351.10 MBs) in 22 sec
Processing rate: 859.4 evts/sec (59.5 MBs/sec)
Close dialog when processing is complete
Show Logs
Rate plot
Memory Plot
Stop
Cancel
Close

Ce qui doit être fait

- Ecrire script de staging des données
  - Cf. cms.prep dans fichier conf xrootd
  - pour pouvoir travailler avec des datasets Proof
- Mesure des performances (e.g. par rapport CAF)
- Espace de stockage pour les sorties des analyses : où, combien, etc… ?