HLT tutorial: Making an HLT trigger

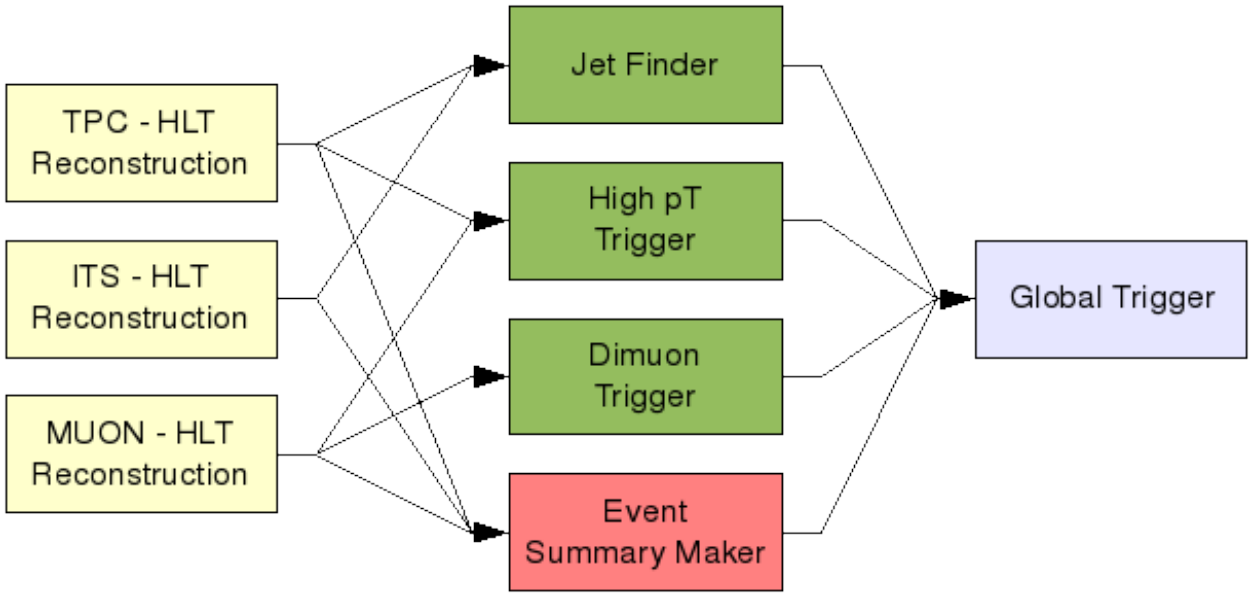
Matthias Richter

Dep. of Physics, University of Oslo

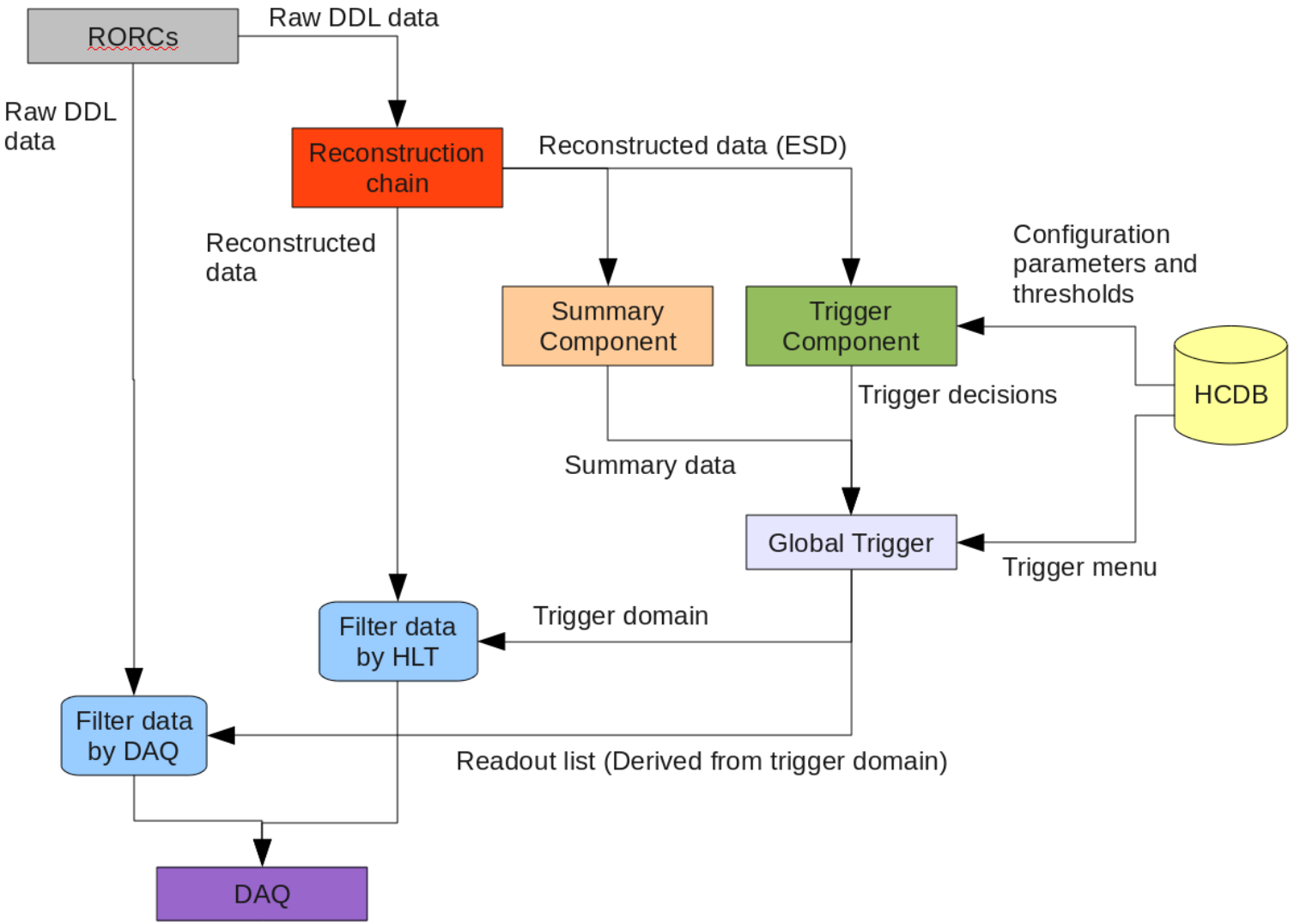
Feb 09 2012

Overview

- Base class for trigger components: `AliHLTTrigger`
- Provides the specific implementation of `AliHLTComponent` → only a subset of the interface needs to be implemented
- Individual trigger components provide input to the GlobalTrigger
- Tiggers provide a *trigger decision* and a *readout list*
- The GlobalTrigger calculates the final trigger decision from the trigger inputs according to a trigger menu



Data flow


RORCs
Raw DDL data
Raw DDL data
Reconstruction chain
Reconstructed data (ESD)
Reconstructed data
Summary Component
Trigger Component
Configuration parameters and thresholds
HCDB
Trigger decisions
Summary data
Global Trigger
Trigger menu
Filter data by HLT
Trigger domain
Filter data by DAQ
Readout list (Derived from trigger domain)
DAQ

Trigger Properties

GetComponentID → GetTriggerName

```
const char* AliHLTTutorialTrigger::GetTriggerName() const
{
  // get name of trigger
  return "TutorialTrigger";
}
```

GetInputDataTypes: optional

```
void AliHLTTutorialTrigger::GetInputDataTypes( AliHLTComponentDataTypeList& tgtList) const
{
  /// inherited from AliHLTComponent: list of data types in the vector reference
  tgtList.clear();
  tgtList.push_back(kAliHLTDataTypeESDObject);
}
```

Spawn: mandatory

```
AliHLTComponent* AliHLTTutorialTrigger::Spawn()
{
  // create instance
  return new AliHLTTutorialTrigger;
}
```

Trigger Processing: DoTrigger

DoEvent → DoTrigger

```
int AliHLTTutorialTrigger::DoTrigger()
{
  // process one event and create trigger information
  if (!IsDataEvent()) {
    IgnoreEvent();  // dont generate any trigger decision.
  }

  int iResult=0;
  int numberOfTracks=-1;

  // read the input data: see next slides

  bool condition=false;

  // here is the trigger algorithm: see next slides

  // add a specific trigger decision object with initialized name
  // the readout list however is fixed
  AliHLTTriggerDecision decision(
                                 condition,
                                 GetTriggerName(),
                                 GetReadoutList(),
                                 GetDescription()
                                 );
  TriggerEvent(&decision);

  return iResult;
}
```

Trigger Processing: AliHLTTriggerDecision

- boolean condition
- name
- readout list
- description

```
root [0] AliHLTReadoutList rolist("TPC")
root [1] AliHLTTriggerDomain domain(rolist)
root [2] AliHLTTriggerDecision decision(1, "TutorialTrigger", domain)
root [3] decision.Print()
Trigger (TutorialTrigger) result = 1
Description = ""
Trigger domain rules (applied in order of first to last):
Include DAQRDOUT:TPC\0:**********
```

Trigger Processing: DoTrigger input data access

Extract ESD object from input, and count tracks

```
// try the ESD as input
const TObject* obj = GetFirstInputObject(kAliHLTDataTypeESDObject, "AliESDEvent");
AliESDEvent* esd = NULL;
if (obj) {
  esd=dynamic_cast<AliESDEvent*>(const_cast<TObject*>(obj));
}

if (esd) {
  numberOfTracks=0;
  esd->GetStdContent();

  for (Int_t i = 0; i < esd->GetNumberOfTracks(); i++) {
    AliESDtrack *esdTrack = esd->GetTrack(i);
    if ( !esdTrack )
      continue;

    numberOfTracks++;
  }
}
```

Trigger Processing: `DoTrigger` trigger algorithm

Compare number of tracks with required condition

Set the trigger description

```
TString description;

if (iResult>=0 && numberOfTracks>=0) {
  if (numberOfTracks>=fMinTracks) {
    description.Form("Event contains %d track(s) with : ", numberOfTracks);
    condition=true;
  } else {
    description.Form("No tracks matching the tresholds found in the central barrel (min tracks %d) with : ",
  }
} else {
  if(IsDataEvent()) {
    description.Form("No input blocks found");
  } else {
    description.Form("No DataEvent found");
  }
}

SetDescription(description.Data());
```

Prepare to run trigger: Make OCDB configuration object

Tool macro HLT/exa/makeComponentConfigurationObject.C

- object path
- configuration string
- OCDB Uri
- run range
- see the macro for documentation or run without arguments

Create HLT/ConfigSample/TutorialTrigger

```
aliroot -b -q -l \
$ALICE_ROOT/HLT/exa/makeComponentConfigurationObject.C'("HLT/ConfigSample/TutorialTrigger", "-mintracks 5", "local://$PWD/OCDB")'
```

Check the result

```
> aliroot -l OCDB/HLT/ConfigSample/TutorialTrigger/Run0_999999999_v0_s0.root
root [0]
Attaching file OCDB/HLT/ConfigSample/TutorialTrigger/Run0_999999999_v0_s0.root as _file0...
root [1] AliCDBEntry->GetObject()->Print()
TObjString = -mintracks 5
```

Prepare to run trigger: Create GlobalTrigger menu

The global trigger menu is compiled at initialization of the GlobalTrigger, see macro for details.

```
aliroot -b -q -l $ALICE_ROOT/HLT/Tutorial/HM-Tutorial.C'("local://$PWD/OCDB")'
```

```
// ////////////////////////////////////////////////////////////////////////////////
// Create the trigger menu.
AliHLTGlobalTriggerConfig config("HM-Tutorial");

config.AddSymbol("domainALLDDL", "AliHLTTriggerDomain", "", "AliHLTTriggerDomain(\"DAQRDOUT:***\\0\")");

// -- Tutorial Trigger
config.AddItem(
"TutorialTrigger",
"domainALLDDL",
"TutorialTrigger"
);


// ////////////////////////////////////////////////////////////////////////////////
// default domain in case there is no global trigger
// readout the output of the reconstruction
// this refers to the domain domainHLTOUT|domainHLTDDL
config.SetDefaultTriggerDescription("No HLT global trigger");

// HLT payload also stored for not triggered events
config.DefaultTriggerDomain().Add("*******", "HLT ");
AliHLTReadoutList readoutlist;
readoutlist.Enable(AliHLTReadoutList::kHLT);
config.DefaultTriggerDomain().Add(readoutlist);
```

Run trigger

```
aliroot -b -q -l \
$ALICE_ROOT/HLT/Tutorial/run-trigger.C \
$ALICE_ROOT/HLT/exa/recraw-local.C'("raw.root","local://$ALICE_ROOT/OCDB", 0, 1, "HLT", "chains=trigger ignore-hltout
loglevel=0x7c")' \
2>&1| tee recraw-local.log
```

```
void run_trigger(const char* uri=NULL, int runno=0)
{
  // set up HLT system to enable configuration registration $
  AliHLTSystem* pHLT=AliHLTPluginBase::GetInstance();
  pHLT->LoadComponentLibraries("libAliHLTUtil.so libAliHLTTrigger.so libAliHLTTutorial.so");

  AliCDBManager::Instance()->SetDefaultStorage(uri?uri:"local://$ALICE_ROOT/OCDB");
  AliCDBManager::Instance()->SetRun(runno);
  AliCDBManager::Instance()->SetSpecificStorage("HLT/ConfigSample/TutorialTrigger", "local://$PWD/OCDB");
  AliCDBManager::Instance()->SetSpecificStorage("HLT/ConfigHLT/HLTGlobalTrigger", "local://$PWD/OCDB");

  // publisher configuration for the binary ESD blocks of writer-conf.C
  AliHLTConfiguration publisher("publisher", "FilePublisher", "", "-datafilelist publisher.txt");

  // TutorialTrigger configuration
  const char* triggerInput="publisher"; // or "GLOBAL-esd-converter"
  AliHLTConfiguration trigger("trigger", "TutorialTrigger", triggerInput, "");
}
```