HLT tutorial: Running an HLT chain in AliRoot

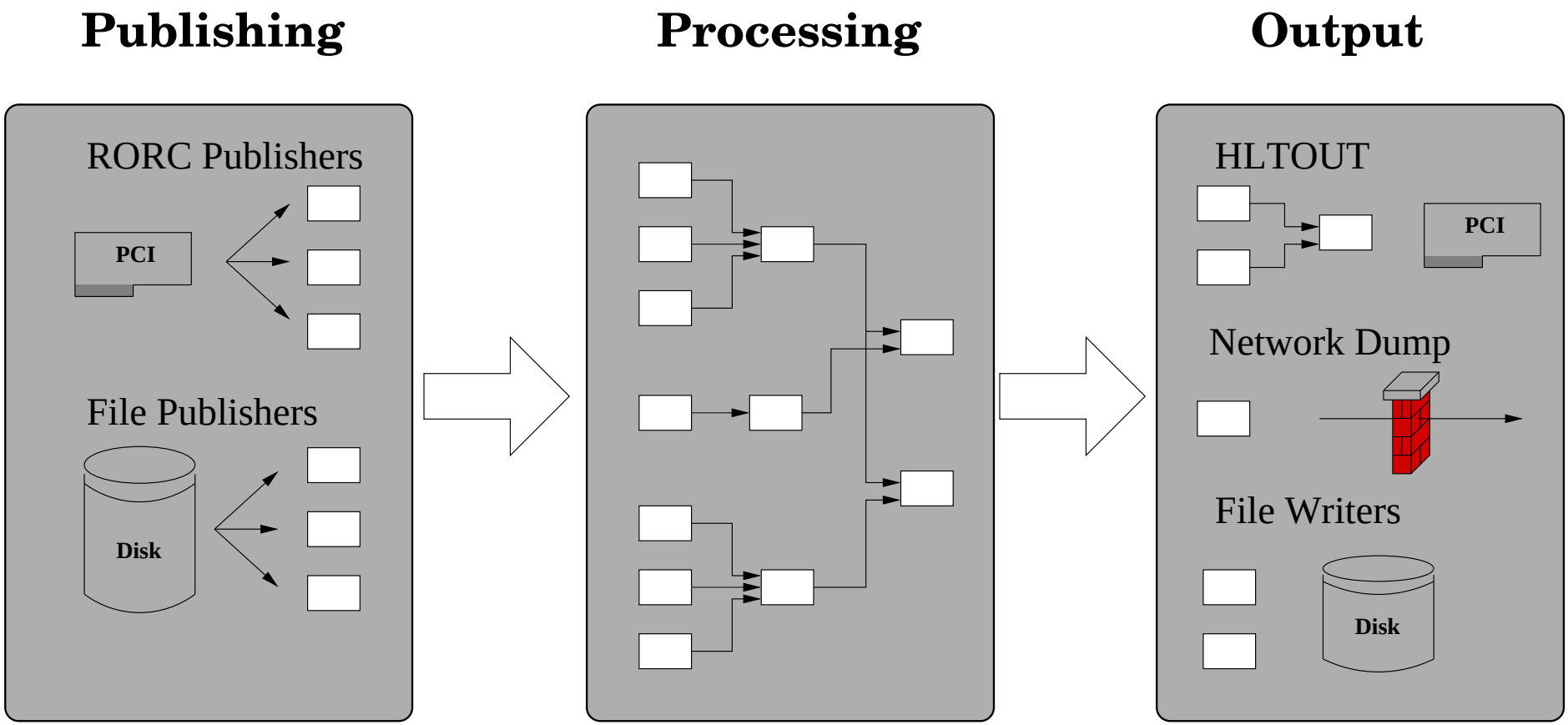
Matthias Richter

Dep. of Physics, University of Oslo

Feb 09 2012

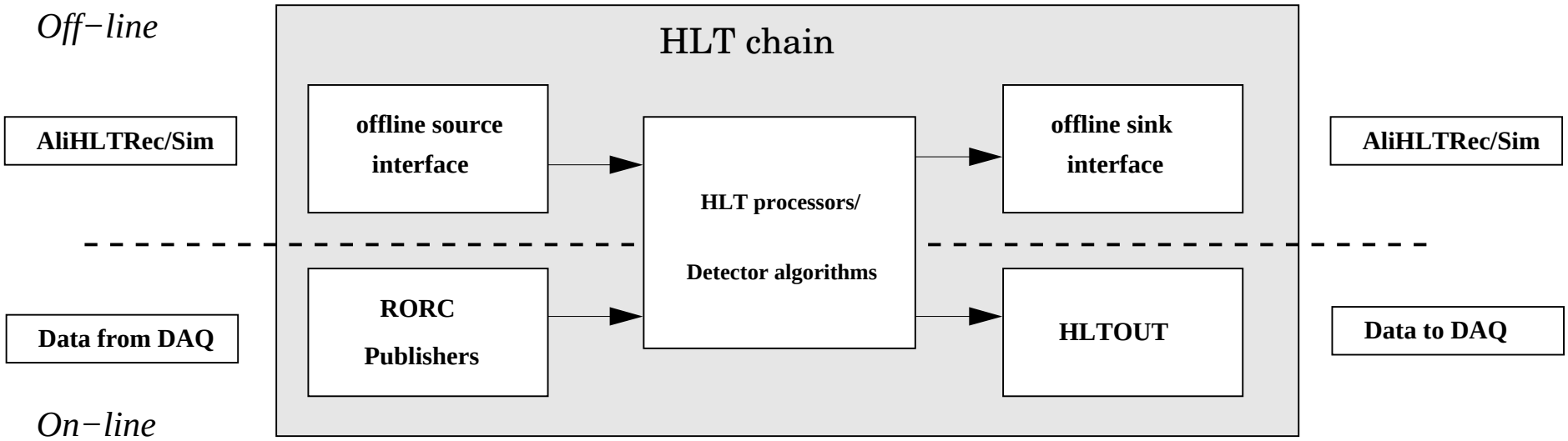
Overview

HLT reconstruction and analysis are organized in *components*, the data flow is established by a *framework* implementation via a dedicated interface



AliHLTSystem

`AliHLTSystem` integrates an HLT chain into AliReconstruction/AliSimulation



Options

- `libAliHLTXXX.so` load component library
- `chains=chain1,chain2` comma-separated list of chain identifiers
- `config=file` read configuration from macro file
- `loglevel=0x7c` loglevel: 0x79 default, 0x7c including info messages

A simple configuration: AliHLTConfiguration

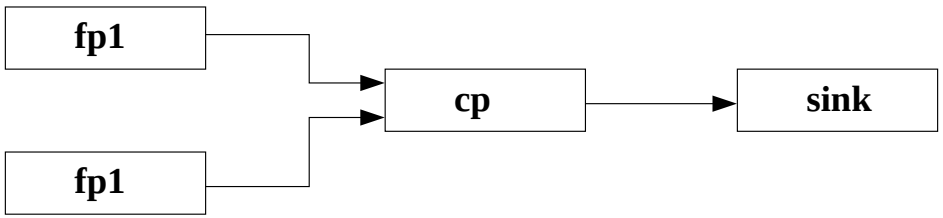
AliHLTConfiguration defines one process with *id, component, parents and arguments* by simply declaring a variable.

```
AliHLTConfiguration c("id", "component", "parents", "arguments")
```

```
{
// publisher 1
AliHLTConfiguration publisher1( "fp1" , "FilePublisher" , NULL,
                                "-datatype DUMMYDAT SMPL "
                                "-datafile data.dat" ) ;
// publisher 2
AliHLTConfiguration publisher2( "fp2" , "FilePublisher" , NULL,
                                "-datatype DUMMYDAT SMPL "
                                "-datafile data.dat" ) ;

// cp subscribes to fp1 and fp1
AliHLTConfiguration copy( "cp" , "Dummy" , "fp1 fp2" , "output_percentage 80" ) ;

// sink subscribes to cp
AliHLTConfiguration sink( "sink" , "FileWriter" , "cp" , NULL ) ;
}
```



Running a simple configuration

Setup AliRoot, Unpack tutorial package and start aliroot

```
cd $ALICE_ROOT
tar xzvf $HOME/tutorial.tgz
cd Tutorial
aliroot -l -b
```

Run chain in AliHLTSystem

```
AliHLTSystem* pHLT=AliHLTPluginBase::GetInstance();
pHLT->ScanOptions("libAliHLTSample.so libAliHLTUtil.so config=simple-conf.C chains=sink");
pHLT->Configure();
pHLT->Run();
```

```
root [0] AliHLTSystem* pHLT=AliHLTPluginBase::GetInstance()
I-AliHLTComponentHandler::AnnounceVersion: ALICE High Level Trigger build on Jan  5 2012 (12:11:30) (embedded A
root [1] pHLT->ScanOptions("libAliHLTSample.so libAliHLTUtil.so config=simple-conf.C chains=sink")
I-AliHLTComponentHandler::LoadLibrary: using libAliHLTSample.so plugin  (gSystem)
I-AliHLTComponentHandler::LoadLibrary: using libAliHLTUtil.so plugin  (gSystem)
(int)0
root [2] pHLT->Configure()
I-AliHLTSystem::BuildTaskListsFromReconstructionChains: custom reconstruction chain: sink
(int)0
root [3] pHLT->Run()
I-AliHLTSystem::ProcessTasks: Event 0 successfully finished (0)
I-AliHLTSystem::PrintBenchmarking: HLT statistics:
    base:                R:0.000s C:0.000s
    input:               R:0.000s C:0.000s
    output:              R:0.000s C:0.000s
    event processing :   R:0.000s C:0.000s
(int)1
```

Running embedded into AliReconstruction

The default reconstruction does not run any HLT chain, the data which are produced on-line are just extracted, but

$\Rightarrow$ HLT chains can be emulated and replace or supplement the data in HLTOUT

HLT Reconstructor Options

```
AliReconstruction reco;
reco.SetInput("raw.root"); // run on raw data // ...
reco.SetOption("HLT", "chains=GLOBAL-esd-converter ignore-hltout");
// ...
```

Note

- HLT emulation in AliReconstruction can only run on raw data, there is no access to detector digits through the AliReconstructor interface
  $\longrightarrow$ see *Running in AliSimulation*
- `ignore-hltout` makes the system to ignore all data blocks stored in the raw data HLTOUT, there might be duplicated data blocks otherwise because of the output of the emulation
- reconstruction has to happen in a *clean directory* which does not contain `galice.root`

AliReconstruction example

Tool macro: recraw-local.C

- Sets all necessary options for AliReconstruction
- Simply to be run from the command line with some options

```
root [0] .x $ALICE_ROOT/HLT/exa/recraw-local.C
recraw-local: Run AliRoot reconstruction locally
 Usage: aliroot -b -q -l \
     recraw-local.C'("file", "cdb", minEvent, maxEvent, modules, hltOptions)'

 Examples:
     recraw-local.C'("alien:///alice/data/2009/.../....root")'
     recraw-local.C'("raw://run12345")'
     recraw-local.C'("raw://run12345", minEvent, MaxEvent)'
     recraw-local.C'("raw.root", "local://$PWD", minEvent, MaxEvent)'

 Defaults
     cdb="raw://"  -> take OCDB from GRID
     minEvent=-1   -> no lower event selection
     maxEvent=-1   -> no upper event selection
     modules="ALL" -> all modules
     hltOption="loglevel=0x7c" -> logging level info and above
```

Assuming some raw data in the working directory, e.g.

```
cd $ALICE_ROOT/test/ppbench/recraw
aliroot -b -q -l $ALICE_ROOT/HLT/exa/recraw-local.C\
         '("raw.root","local://$ALICE_ROOT/OCDB", 0, 5, "HLT", "chains=GLOBAL-esd-converter ignore-hltout")' 2>&1\
         | tee recraw-local.log
```

recraw-local.C with custom chain

Define a configuration with a *FileWriter* component.

- parent defined by command line argument
- files written to directory *data*, each event in a subdirectory
- a configuration file *publisher.txt* is written containing the configuration for a *FilePublisher* component to publish exactly this sequence

HLT/Tutorial/writer-conf.C

```
void writer_conf(const char* parent)
{
  // set up HLT system to enable configuration registration
  AliHLTSystem* pHLT=AliHLTPluginBase::GetInstance();

  // writer configuration
  AliHLTConfiguration writer("writer", "FileWriter", parent,
                             "-publisher-conf publisher.txt -directory data -subdir");
}
```

Run the configuration macro before recraw-local.C

```
aliroot -b -q -l $ALICE_ROOT/HLT/Tutorial/writer-conf.C'("GLOBAL-esd-converter")' \
$ALICE_ROOT/HLT/exa/recraw-local.C'("raw.root","local://$ALICE_ROOT/OCDB", 0, 5, "HLT", "chains=writer ignore-hltout")' \
2>&1| tee recraw-local.log
```

HLT simulation

HLT Simulation is included in AliSimulation through the class AliHLTSimulation

HLT Simulation Options

```
AliSimulation sim;
// ...
sim.SetRunHLT("chains=GLOBAL-esd-converter,GLOBAL-Trigger");
// ...
```

Note

- HLT simulation runs at the end of AliSimulation and produces the HLT.Digits.root
- Internally AliHLTSystem runs in the same way as in Reconstruction, however, input components might be different
- HLT simulation is the only way to simulate an HLT chain on the detector digits, i.e. with MC information
- If raw data generation is activated, the HLT chain is run twice in two separated systems
  - first on digits, the HLT.Digits.root file contains the output of HLT simulation including MC information
  - on generated detector ddl files, the HLT ddl file contains the output of HLT simulation on raw data

HLT simulation example

Tool macro: simhlt.C

- Works on already simulated data sample
- Switches off event generation and detector simulation
- Simply to be run from the command line with some options

```
root [0] .x $ALICE_ROOT/HLT/exa/simhlt.C
 Usage: aliroot -b -q -l \
     aliroot -q simhlt.C'("hltoption", "rawdataoptions", nofEvents, runNo, "cdbUri")'

 Examples:
     aliroot -q simhlt.C'("loglevel=0x7c chains=GLOBAL-esd-converter")'
     aliroot -q simhlt.C'("loglevel=0x7c !libAliHLTITS.so")'
     aliroot -q simhlt.C'("loglevel=0x7c rawfile=")'

 Defaults
     rawdataoptions=""          -> skip generation of raw data
     nofEvents=-1               -> take event count from simulated sample
     runNo=-1                   -> take run number from simulated sample
     cdbUri="local://$ALICE_ROOT/OCDB"

 The definition of the chain depends on the availability of simulated raw
 data. If raw data is available, the HLT chain runs on this, otherwise on
 the digit data. NOTE: propagation of MC information is only possible
 for the latter case. Raw data generation must be either switched of, i.e.
 rawdataoptions is empty, or explicitly explicitly ignored by the HLT
 option 'rawfile=' (note the empty argument to the option).
```

Input and output

- Standard handling for ESD objects in HLTOUT
- Output blocks from the emulation are included in the standard HLTOUT handling
- Several publisher and writer components in `libAliHLTUtil.so` provide data sources and sinks for the HLT offline system

- *FilePublisher*: publisher for binary files
  `HLT/BASE/util/AliHLTFilePublisher`
- *AliRawReaderPublisher*: publisher for raw files through AliRawReader
  `HLT/BASE/util/AliHLTRawReaderPublisherComponent`
- *AliHLTOUTPublisher*: publisher for binary blocks from HLTOUT
  `HLT/BASE/util/AliHLTOUTPublisherComponent`
- *ESDMCEventPublisher*: publisher for ESD objects
  `HLT/BASE/util/AliHLTESDMCEventPublisherComponent`
- *FileWriter*: writer for binary blocks
  `HLT/BASE/util/AliHLTFileWriter`
- *ROOTFileWriter*: writer for ROOT objects
  `HLT/BASE/util/AliHLTRootFileWriterComponent`