Evolution of HLT infrastructure

athenaMT/PT $\rightarrow$ athenaHLT

*(**P**esa) **S**teering **C**ontroller*

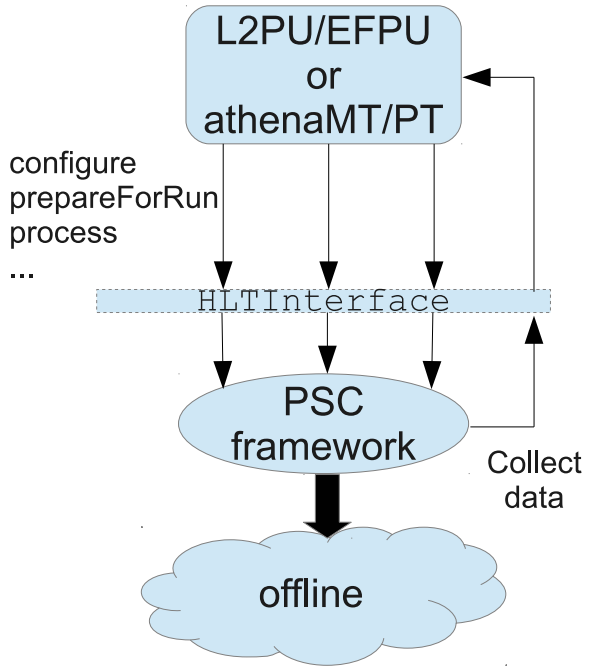
Ricardo Abreu
Frank Winklmeier
Werner Wiedenmann

CERN

April 5, 2013

Context

- Much of the trigger code comes from offline
  - depends on offline framework
- But the purpose is to run online, i.e. inside l2pu/efpt instead of an athena process
  - to have the Athena/Gaudi framework available, something is needed to act as a glue between online and offline code
- Main place of integration between HLT and TDAQ → achieved with HLTInterface
  - package that defines API that the PU can use and the HLT has to implement



PSC framework

There is a package called TrigPSC but I am referring to the packages under `$SVNROOT/HLT/Trigger/TrigControl`

Evolution of the HLTInterface

With the evolution of the dataflow there is a new HLTInterface

Main changes so far:

1. Merging of the L2 and EF – PSC integrates HLT in 2 distinct online processes (L2/EF) ⇒ needs to be aware of distinctions ⇒ concepts that permeate the code everywhere
2. Parameters received through boost property trees, including configuration – No more OKS – considerable re-writing of configuration step
3. Data collection – as presented by Werner

Pending changes

1. New internal transition after prepareForRun – Needed for individualization after PU forking
2. Reworking of user commands – to be discussed...

These are only at the level of the HLTInterface, but the interface does not specify everything (e.g. what goes inside the ptrees)

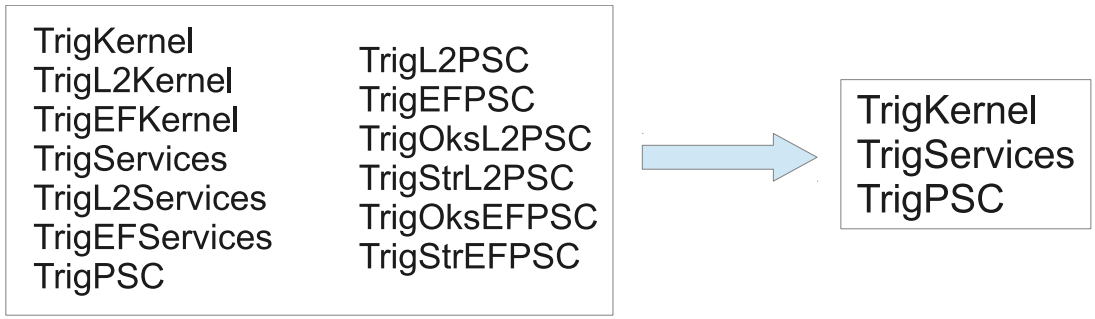
Impact – PSC

Changes on the HLTInterface of course affect both sides

Impact on the PSC

Impact on the code varies from simple string substitutions or parameter updates to class model and package structure reorganization.

Example with core TrigControl packages:



- Some disappear because the functionality is replaced (e.g. TrigOks*XX*PSC and TrigStr*XX*PSC)
- Most disappear because of the distinction between L2/EF disappears, but the functionality is maintained and moved (e.g. Trig*XX*PSC)

Impact – athenaHLT

If the PSC wraps the offline code and gives it an online interface, athenaHLT wraps that online interface and executes it offline!

Impact on athenaHLT

- merge athenaMT/PT $\rightarrow$ athenaHLT $\iff$ use/comply to new HLTInterface
- There are mainly four different levels that are affected
  1. At the bottom, the actually calls to the HLTInterface
     - light changes
  2. Above that, the C++ to Python bindings
     - e.g. ptree type, interface itself, exceptions (?)
  3. The filling in of the ptrees
     - main impact is on the configuration transition
  4. user interface

The changes will eventually extend beyond the impact of the HLT interface:

- monitoring
- other dataflow impacts – the dataflow system has changed a lot and athenaHLT has to emulate everything that is needed for a standalone HLT run

Conclusion

Impact on both sides of the HLTInterface

Special care needed when editing the code because:

- Many changes
- Difficult to test anything before everything is in place
- Accommodate different running conditions
  - L2 or EF
  - athenaMT or athenaPT
    - batch or interactively mode
- Substantial configuration changes:
  - method (oks/string/ptree)
  - target (EF/L2 $\rightarrow$ HLT)
  - contents, defaults

Also, a lot of testing will be needed. A dedicated release will probably have to be setup.