Status of muon data conversion and muon HLT trigger in val (ex.mig2) nightlies

s. spagnolo - Lecce

Muon Data Conversion

general ideas:
data conversion embedded in AlgTools used both in offline and online

XxxRawDataProviderTools perform BS-> RDO
XxxRdoToPrepDataTools perform RDO->PRD

in the offline running on BS
1st reco.step: XxxRawDataProvider algos running XxxRawDataProviderTools on the whole event (all possible ROB fragments)
2nd reco. step: XxxRdoToXxxPrepData algos running XxxRdoToPrepDataTools on the whole event

in the offline running on MC RDO
1st reco. step: XxxRdoToXxxPrepData algos running XxxRdoToPrepDataTools on the whole event

Muon Data Conversion

in the online (BS input)
LVL2 algos use XxxRawDataProviderTools to obtain input data, i.e. RDO collections, in the RoI region
EF algos use XxxRdoToPrepDataTools to obtain input data, i.e. PRD collections in the RoI region

in the online emulation on MC RDO
LVL2 algos skip XxxRawDataProviderTools to read directly input data, i.e. RDO collections, in the RoI region from POOL
EF algos use XxxRdoToPrepDataTools to obtain input data, i.e. PRD collections converting RDO collections in the RoI region

Muon Data Conversion

in the online (BS input)
EF algos use XxxRdoToPrepDataTools to obtain input data, i.e. PRD collections in the RoI region:

Therefore XxxRdoToPrepDataTools need to behave differently according to input type: BS or POOL RDO

TO DO:

- EF algos have to use XxxRdoToPrepdataTools to ask for PRD collections (instead of retriving PRD container from SG and look for requested collections)
- add a job-opt flag bool inputBS to all XxxRdoToPrepDataTool
- define a job-opt fragment that checking job input type sets inputBS flag
- use in XxxRdoToPrepDataTool inputBS to decide is
  - true: request a vector of RDO collections to the XxxRawDataProviderTools
  - false: get the requested Rdo collections from the Container retrieved from SG (current behaviour)

Muon Data Conversion

Current difficulties:

XxxRawDataProvider tools do not have a interface with a vector of RDO collections

they have an interface with:

- ROB fragment pointer
- vector of RDO collections

This implies that the user (XxxRdoToPrepdataTools) would need to

- depend on RobDataProvider
- ask the cabling svcs what ROB ID contains the Rdo collectiosn requested
- ask the RoBDataProvider to get the pointers to the requested ROBids
- ask the RawDataProvider Tools the vector of RDO colls. by passing the ROB fragments

Muon Data Conversion

My proposal:

extra interface in XxxRawDataProvider tools accepting only the vector of RDO collections;

the implementation:

- ask the cabling svcs what ROB ID contains the requested Rdo collections
- ask the RoBDataProvider to get the pointers to the requested ROBids
- use the existing interface with input arguments the vector of RDO colls. AND the ROB data fragments

This implies that

- the XxxRawDataProvider depend on the ROBdataProvider
- all the BS data access is embedded in the XxxRawDatrtaProviderTools and, hence, hidden to High Level clients, like the rdoToPrepDataTools
- the LVL2 can keep using the existing interface

Muon Data Conversion

My proposal BUT ALSO THE CURRENT schema
need extra time (~1 week) for implementing the RoI driven data access for the EF

Fall-back solution:
in the EF jobs runs before the steering the
offline XxxRawDataProviderAlgos + XxxRdoToPrepDataAlgos if BS input
offline XxxRdoToPrepDataAlgos if MC RDO input

The plan:
- start testing extensively the fall-back solution
- start implementing the schema RoI driven data access in the XxxRdoToPrepdataTools and XxxRawDataProviderTools (14.2.10 should wait for that)
- can we stage for bug-fixes on 14.2.10 the EF algo changes ?