Pre-DC2 Refactoring Proposal

Status bits

Storage of data producer names instead of status bits

Starting at CalClusters level, objects are less numerous in the event (as compared to lower level objects) and can be a little bigger. This is why we would prefer to store the producer name as a string in each CalCluster and in each CalCorToolResult, rather than rely in status bits. Status bits are predefined and can hardly be used with new/user algorithms, thus it reduces flexibility.

To be removed:

- CalCluster::*CLUSTER
- CalCorToolResult::CALVALS, PROFILE, LASTLAYER, RAWENERGY

Worth to say: the use of this producer name by clients can be handy and even necessary in some situation, but should be avoided whenever possible, since this is restraining the global flexibility of the system.

IMPLEMENTED. READY TO COMMIT.

Avoid algo-oriented status bits

We should avoid status bits saying things about algorithms, because this reduce our freedom with the way we arrange and change those algorithms. We should prefer eventually status bits which say things about the "state" of the data.

To be removed :

- CalCluster::MIPTRACK
- CalCluster::MOMENTS
- CalCluster::ENERGYCORR
- CalCluster::MIPFIT

IMPLEMENTED. READY TO COMMIT.

Generalize the validity status bits.

Is is safe to use status bits so to flag the data which is valid or not.

To be added :

- CalCluster::AXIS (instead of MOMENTS) ***TO BE DONE.***
- CalEventEnergy::VALIDPARAMS ***IMPLEMENTED. READY TO COMMIT.***

Etc.

It seems CalCluster::ISOLATEDCLUSTER is redundant, and equivalent to !ALLXTALS.

To be removed:

- CalCluster::ISOLATEDCLUSTER

For the remaining useful status bits (for example the ones saying if this or that data is valid), the methods signatures could be made more uniform in the various CalRecon classes.

IMPLEMENTED. READY TO COMMIT.

Info Fillers

Do we really need several info fillers ? Or event a list of them to be applied at each event ? Can we better factorize the code between the current two fillers ? Should we append the filler name to the CalCluster producer name ?

UNDER WORK

Correction Tools

Kind of energy

It seems not all the tools are returning the same kind of energy. Most of them are returning the evaluation of the total gamma energy. We must investigate if CalValsCorrTool, which apparently returns the cal-only energy, can be modified so to return the same as other tools, and we must also investigate if Tkr code can be modified accordingly (and what about CalRawEnergyTool ?).

TO BE DONE.

Handling of multiple clusters

Debate let us think that the tools should not process each cluster individually, but should receive the complete list of available clusters. They can then consider only the front/best one if they wish.

IMPLEMENTED. READY TO COMMIT.

CalTransvOffsetTool

It seems it is not used. As a first step, we want to remove it from the list of default tools. What's more, it is not computing an energy. If we must keep it, this should certainly not be a kind of ICalEnergyCorr.

IMPLEMENTED. READY TO COMMIT.

CalEventEnergy

No more singleton

We want to enable multiple instances of CalEventEnergy, so that one make several differetn interpretations of the event, for example 1 Gamma and 2 PI0. As an additionnal benefit, this will ease the management of CalEventEnergy in the TDS and in the ROOT trees.

IMPLEMENTED. READY TO COMMIT.

CalEventEnergyAlg passes

The current implementation is more or less hard coding that there are two passes, and select the best energy candidate depending on the pass. We propose something more flexible : there is no information about the pass, and the best energy candidate is selected depending on the ones which are present in the list : energy of CalValCorrTools if it is present, and CalRawEnergyTool otherwise.

To be removed :

- CalCluster::PASS_ONE
- CalCluster::PASS_TWO

IMPLEMENTED. READY TO COMMIT.

A detail: why duplicate the data fo the best CalCorToolResult ? Why not simply point to it ?

UNDER WORK