Subject: LCIO: first trial and first questions...
Posted by antonio.bulgheroni on Thu, 01 Feb 2007 10:42:39 GMT
View Forum Message <> Reply to Message

Dear all,
    since we have this brand new forum I'm forwarding here a very useful discussion among Frank and me about the use of LCIO in our work.

Unfortunately the discussion started when this forum wasn't available yet so all messages will appear as sent by me! Sorry for that!

Your contribution is very much appreciated!

Cheers,

Antonio

Subject: Re: LCIO: first trial and first questions...
Posted by antonio.bulgheroni on Thu, 01 Feb 2007 10:45:26 GMT
View Forum Message <> Reply to Message

Ciao Frank,
sorry for disturb you, but it's my very first experience with this completely new world of LCIO + MARLIN et al.

First of all, I have installed all the needed (and optional) tools. BTW, I've installed LCIO v01-08 since it was advertised as the latest stable version on ilcsoft web page, but I've the feeling this is not really the latest version. Then as a second step I'm training with the use of LCIO. For that reason I tried to write a very simple code to translate a test MIMOSA V file (with each pixel output written in ASCII row-wise). This test file contains 100 events.

The idea is the following. I created a collection called matrix of LCIO::TRACKERRAWDATA. I push_backed to this collection a TrackerRawData, one for every pixel in the detector. I added the collection to the event with the name "raw-matrix" and then I wrote the event via the LCWriter. It looks reasonable to me, doesn't it? This is what I did in the converter.cc attached file.

So far, so good. The problem comes when I try to read the file back. Using the dumpEvent tool, I got exactly what I would expect, but one single strangeness.

COLLECTION NAME   COLLECTION TYPE   NUMBER OF ELEMENTS

```
===============================================================
=
raw-frame              TrackerRawData              0
---------------------------------------------------------------
```

The "number of elements" is zero. The collection seems to be empty. To cross-check, I tried also to add another collection (of LCIO::SIMCALORIMETERHIT) as you did in the documentation. This is the
"// ** //" commented part of the converter.cc file. Doing so, what I got from the dumpEvent is the following:

```
---------------------------------------------------------------
COLLECTION NAME         COLLECTION TYPE      NUMBER OF ELEMENTS
===============================================================
=
raw-frame              TrackerRawData              0
test           SimCalorimeterHit       262144
---------------------------------------------------------------
```

being 512 x 512 = 262144 the number of pixels in the detector.

What's wrong? Could you have a look at the attached code?

For the time being, I'm using a TrackerRawData to describe one single pixel, but probably it is wiser to use TrackerRawData to describe a full detector even if I cannot use any longer the cellID's to specify the pixel coordinates. Those can, anyway, be retrieved from the order the pixel appears into the adcValue STL vector as it actually is in the sucimaPix framework. What do you think about that?

That's all for now and thank you very much for your help.

Cheers,

Antonio

File Attachments
1) converter.cc, downloaded 874 times

Subject: Re: LCIO: first trial and first questions...
Posted by antonio.bulgheroni on Thu, 01 Feb 2007 10:47:16 GMT
View Forum Message <> Reply to Message

[Reply from Frank]

Hi Antonio,

from a quick look at your code I didn't find any obvious problems.
Could you please send me a (small) input test file so I can run it
for further investigation.

 Thanks, Frank.

Subject: Re: LCIO: first trial and first questions...
Posted by antonio.bulgheroni on Thu, 01 Feb 2007 10:48:48 GMT
View Forum Message <> Reply to Message

yes... sure... It is attached to this mail. It's just a one event file. If you want the full 100 event file I
can put it somewhere on the web.

Thanks a lot!

toto

File Attachments
1) test.dat, downloaded 838 times

Subject: Re: LCIO: first trial and first questions...
Posted by antonio.bulgheroni on Thu, 01 Feb 2007 10:51:02 GMT
View Forum Message <> Reply to Message

Dear Frank, just to add another little piece of information...

I've tried to printout the size of the collection cout << matrix->size() << endl; just before and after
the call to the lcWriter and in both cases I got the correct value of 262144. So I guess there is
something wrong with the IO implementation.

cheers,

toto

Subject: Re: LCIO: first trial and first questions...
Posted by antonio.bulgheroni on Thu, 01 Feb 2007 10:54:46 GMT
View Forum Message <> Reply to Message

[Reply from Frank]

Hi Antonio,

sorry for the somewhat delayed reply - there where actually a number of issues with your code
and LCIO that I had to figure out.

 First of all you hit a real bug in the underlying I/O code (the
sio-library) as you where suspecting. Something goes wrong with the
padding for shorts (sio uses xdr which only knows 32 bit words) which in
turn causes corrupted files if one tries to store an odd number of shorts
(the ADC values). Simply never happened as the TPC people allways store a
full waveform which has a power of 2 number of entries and all other data
items are either chars or 32-bit quantities...
Anyway I will provide a patch as soon as I have a fix for the problem.
 Apart from the fact that of course one should be able to store also odd
numbers of ADC values and one in particular it is probably not a good idea
to do so for the non-zero-suppressed pixelsensor raw data. It simply
causes an awfully large overhead. Here I would suggest that you store a
full matrix (or parts of it - see next point) in one TrackerRawData
object. Otherwise file sizes will be very large and I/O unnecessary slow.
I modified the example accordingly.
 Strangly the sio-library refused to store more than 2^16 short values in
one call. This of course also needs to be understood and fixed. Anyway for
the time being one needs to restrict the pixels per object to 65536.
In the example code I divided the 512X512 sensor into four quarters (note
that the code won't work for larger sensors of course!) and encoded the
pixel numbers/ranges in the cellid. (please use the attached header file
at $LCIO/src/cpp/include/UTIL/CellIDEncoder.h until we release a
proper LCIO patch).
 you cannot use special characters in LCIO collection names - so I
changed 'raw-frame' to rawframe'
 I changed the logic in the example program, so that it keeps reading ADC
values from a file into a buffer of 512x512 shorts and creates a new event

for every complete sensor read

I will try and provide a patch for LCIO asap so that you can continue to develop code for the pixel sensors.

 Cheers, Frank

File Attachments
1) converter(2).cc, downloaded 959 times

Subject: Re: LCIO: first trial and first questions...
Posted by antonio.bulgheroni on Thu, 01 Feb 2007 11:00:15 GMT
View Forum Message <> Reply to Message

Ciao Frank,
     first of all thank you very much for the very quick turn around. You shouldn't apologize for being in late with Italians, they are used to much much longer waits!

Quote:
First of all you hit a real bug in the underlying I/O code (the sio-library) as you where suspecting. Something goes wrong with the padding for shorts (sio uses xdr which only knows 32 bit words) which in turn causes corrupted files if one tries to store an odd number of shorts (the ADC values). Simply never happened as the TPC people allways store a full waveform which has a power of 2 number of entries and all other data items are either chars or 32-bit quantities...
Anyway I will provide a patch as soon as I have a fix for the problem.

Ok... now it is a feature...

Quote:
Apart from the fact that of course one should be able to store also odd numbers of ADC values and one in particular it is probably not a good idea to do so for the non-zero-suppressed pixelsensor raw data. It simply causes an awfully large overhead. Here I would suggest that you store a full matrix (or parts of it - see next point) in one TrackerRawData object. Otherwise file sizes will be very large and I/O unnecessary slow.
I modified the example accordingly.

Strangly the sio-library refused to store more than 2^16 short values in one call. This of course also needs to be understood and fixed. Anyway for the time being one needs to restrict the pixels

per object to 65536.
In the example code I divided the 512X512 sensor into four quarters (note that the code won't work for larger sensors of course!) and encoded the pixel numbers/ranges in the cellid. (please use the attached header file at $LCIO/src/cpp/include/UTIL/CellIDEncoder.h until we release a proper LCIO patch).

So for the time being I'll have a look at the CellIDEncoder and I'll need to adapt it to a full Mimosa readout (1024 x 1024).

Quote:
you cannot use special characters in LCIO collection names - so I changed 'raw-frame' to rawframe'

Ok!

Quote:
I changed the logic in the example program, so that it keeps reading ADC values from a file into a buffer of 512x512 shorts and creates a new event for every complete sensor read

That's something I buy, this is also the way I usually do it. In the case of an ASCII file, it is a real advantage, but when the input file is a formatted binary, then you can read one full event with just one I/O operation.

Quote:
I will try and provide a patch for LCIO asap so that you can continue to develop code for the pixel sensors.

In the mean time, I'll exercise a bit with Marlin and I can implement the converter as a subclass of DataSourceProcessor.

Thanks again....

ciao ciao

toto

Subject: A Marlin processor for pedestal calculation
Posted by antonio.bulgheroni on Thu, 01 Feb 2007 14:14:38 GMT
View Forum Message <> Reply to Message

Dear all,
   as promised, I'm trying to write a first attempt of a Marlin processor able to calculate pedestal and estimate noise from a LCIO input file containing pixel detector data.

Since I hope that in some time (see this thread) we will be able to save a full detector (when I say detector I mean one telescope plane) output in a TrackerRawData structure, I'm basing the processor on the idea that the input LCIO file will contain for each event a collection of TrackerRawData's, one for each detector plane.

In my standard analysis procedure, I do the following:

 In a loop over all events (or a user selected subset, say the first 100) I calculate the first approximation to pedestal and noise using the average and the variance of each pixel signal distribution. In this first loop, I usually do not apply neither common mode suppression nor hit rejection algorithms, but I flag very noisy pixel as bad so that they are not considered in the following analysis steps.
 The first order correction to the pedestal and noise values is obtained in a second loop on events in this way:

 Since we already have an indication about each pixel noise, for each event we can reject hit pixels on a signal to noise ratio base.
 For each event the common mode level is calculated and consequently a new pedestal value is calculated.
 Again, at the end of the event loop, I flag other too noisy pixels. Usually this second bad pixel masking is much weaker than the first.

 The previous point can be reiterated several times depending on the residual common-mode level. Usually not needed.

So my first idea was to create a PedeNoiseProcessor doing the full procedure. But how to convince Marlin to loop at least twice on all events? Is it possible?

The second idea, probably easier and (maybe) more elegant is to write a PedeNoiseProcessor that is just doing point 1. And then write a second processor, say CommonModeSupProcessor that, using some initial pedestal and noise value, produces the second order approximation. In this way, one can decide how many time the common mode sup algo should be applied just adding CommonModeSupProcessor's to the steering file. Do you think is doable?

Now, I have a more general question on how the framework works. Consider a steering file in

which three processors (processorA, processorB and processorC) are active and properly configured. How is Marlin working? It is something like:

Case 1

```
for (iEvt = 0; iEvt < nEvt; iEvt++) {
   processorA(iEvt);
   processorB(iEvt);
   processorC(iEvt);
}
```

or something like

Case 2

```
for (iEvt = 0; iEvt < nEvt; iEvt++) processorA(iEvt);
for (iEvt = 0; iEvt < nEvt; iEvt++) processorB(iEvt);
for (iEvt = 0; iEvt < nEvt; iEvt++) processorC(iEvt);
```

If it is like in case 1, my idea is not feasible, because I need to loop on all event with PedeNoiseProcess before starting with CommonModeSupProcess.

Thanks, for your help

Antonio

Subject: Re: LCIO: first trial and first questions...
Posted by gaede on Fri, 02 Feb 2007 10:35:09 GMT
View Forum Message <> Reply to Message

Hi Antonio et al,

I made a new tag in LCIO: v01-08-01-pre (a prerelease patch version) that fixes the issues that we encountered two days ago:

proper collection names are enforced (need to be valid C/C++ names - no special characters

except '_')

 there actually was _no_ bug in the I/O routine for odd numbers of shorts (e.g. one ADC value) this was a very strange side effect with the invalid collection name 'raw-frame'

so in fact the code that you sent me originally runs fine if you change:

148c148
<   event->addCollection(matrix,"rawframe");

>   event->addCollection(matrix,"raw-frame");

(except for the logic of reading the data and creating events - one additional event is created)

 as mentioned previously, of course one should not use TrackerRawData to store single pixels if non-zero-suppressed for performance reasons

 a bug in the I/O routine of SIO that limited the number of shorts to 65k has been fixed - now there is no more limit in the number of ADC values that can be stored in a TrackerRawData object

-> a full sensor data matrix can be stored in one object (no need to split into subquadtrants or the like)

 CellIDEncoder has been fixed to work with Tracker(Raw)Data and TRackerPulse - also dumpevent will print the cellid words if the CellIDEncoder has been used for encoding (or a proper decoding string has been written manually)
-> note that there is no need to store x and y pixel ranges any more, however you might want use this feature to make object sizes smaller sth. the group should decide on

 I tested the code under SL3, SL4 and OSX 10.4 (intel) but of course it should also run under other common linux distros

Let me know of any further issues or questions that come up.

 Cheers, Frank.

Subject: Re: LCIO: first trial and first questions...

Posted by antonio.bulgheroni on Fri, 02 Feb 2007 12:57:40 GMT
View Forum Message <> Reply to Message

Dear Frank,
    thanks a lot for your reply and all the work you did. I'll download and install this patch. In the mean time I'll continue my experiments with Marlin.

Thanks again!

Antonio

Subject: Re: A Marlin processor for pedestal calculation
Posted by gaede on Fri, 02 Feb 2007 17:35:27 GMT
View Forum Message <> Reply to Message

Hi Antonio,

the event loop in Marlin right now is really strictly linear, i.e. every event (up to maxRecord) is presented to every processor in the order they are defined in the steering file - similar to what you call 'case 1'. This is the natural choice for normal reconstruction/analysis where you add additional information to the event with every processor.

By using conditions in the execute section one can influence the porogram flow, e.g. only call certain part of reconstruction or analysis if certain pre-conditions are fullfilled. One can also of course have different sets of processors for different events, e.g. calibration and recontruction. But in principle every event will be processed exactly once by one 'path' of processors.

If your calibration will run on the full data sample of the job you can of course specify the set of input files twice and use a flag (isCalibration=true) for the first set and a flag (isCalibration=false) for the second set and so achieve what you want.

However if you want to run a calibration typically on the first N events and then process the full, larger set with these constants   one needs to 'rewind' the data set when the calibration is finished and then start from the beginning for the rest of the processing.

I have written some experimental code that would allow to do so.
Please check out cvs tag fg20070202-rewind and look at $MARLIN/src/Testprocessor.cc.
When using the attached steering file the processor loops three times over the first three events for 'calibration' and then the rest (in this case only output processor) is called for all events. If this works for your case of pedestal calibration we could make a relaese of this 'feature' soon.
It migh be usefull also for other groups that want to do 'injob calibration'

Cheers, Frank.

  Cheers, Frank.

File Attachments
1) test.xml, downloaded 1831 times

Subject: Re: A Marlin processor for pedestal calculation
Posted by antonio.bulgheroni on Mon, 05 Feb 2007 10:59:46 GMT
View Forum Message <> Reply to Message

Dear Frank, thanks for your reply.

I like this idea of rewind the event loop and I believe it is going to be helpful also in many other cases, for example for the eta function generation as presented by Auguste. I will try as soon as possible to modify my PedeAndNoiseProcessor in order to use this feature.

Now the question is the following: usually one has a specific pedestal run (where in principle no particles are hitting the telescope). This run may be used by several other "real" runs as a starting point for pedestal and noise. So the typical idea is to

 calculate pede and noise at the beginning,
 save it into a separate file and
 re-use it whenever you need.

To do that, we can imagine to have a steering file that exploiting the rewind feature is looping as many time as needed on the input data to produce the pedestal file that it is actually containing one event only with a TrackerData for pedestal and one for the noise. Then a second steering file in which a ApplyPedestalCorrectionProcessor is loading from the pedestal file the needed calibration info and apply the correction on a event base. This requires a processor handling I/O operation from a different file. Do you believe is it possible?

I would like to hear comments also from the rest of the (EUDET) community...

Cheers,
Antonio

Subject: RunHeader EUDET specific parameters
Posted by antonio.bulgheroni on Mon, 05 Feb 2007 11:33:48 GMT
View Forum Message <> Reply to Message

Dear all,

I'm trying to figure out which are the specific parameters we would like to save within the LCIO RunHeader.

The LCIO data model allows the user to add "named" parameters of integer, float and string type to the run header, the event and to each collection. The basic underlying idea is to store parameters into a map and use the name key to retrieve them.

Without breaking any LCIO rules, I've prepared a class EUTelRunHeaderImpl with no data members but with methods to get/set our specific parameters into the run header. For example, instead of writing:

runHeader->setValue("DAQSWVERSION",1.0293);
float softVersion = runHeader->getFloatVal("DAQSWVERSION");

one can use

runHeader->setDAQSWVersion(1.0293);
float softVersion = runHeader->getDAQSWVersion();

This may be very useful when the number of parameters grows up and we have always to remember the correct parameter name to set/get it. I believe that this is especially true for people involved in the DAQ software. If you don't agree, you can also not use it !!!

Now, here comes a list of parameters I've already implemented for the run header:

 RunHeaderVersion (float)
 DateTime (string) a time stamp in human readable format
 DataType (string) a flag to indentify real DAQ data or simulated or converted...

 DAQHWVersion (float) It represents the version of the hardware part of the DAQ.
 DAQHWName (string) It is the name of the DAQ HW system. In principle we can have many!

DAQSWVersion (float)
DAQSWName (string) as above but now for the software part of the DAQ

SimulSWVersion (float)
SimulSWName (string) as for the DAQ but here for the simulation software used for data generation
 GeoID (int) an identification number to identify the run geometry within the geometry database
 NoOfDetector (int) this is the number of detectors in the current run. It is redundant since this information is also stored within the geometry db, but at least for on-line monitoring I believe that access to an external database shouldn't be compulsory. This number may also not corresponds to the number of planes.
 MinX, MaxX, MinY, MaxY (vector<int>). To properly store data into a Tracker(Raw)Data object we have to use the CellIDEncoder. This requires to know the first and the last pixel of each sensors along both directions. MinX/Y may be different from zero, when there are multiple sensors in a plane, or when we consider a detector one channel (as in Mimostar2).

Should we add some standard field about the DUT? Adding other parameters is always possible.

Please comment this list and add what is (for sure) missing!

Cheers,

Antonio

Subject: Re: A Marlin processor for pedestal calculation
Posted by antonio.bulgheroni on Mon, 05 Feb 2007 13:42:12 GMT
View Forum Message <> Reply to Message

gaede wrote on Fri, 02 February 2007 09:35
Please check out cvs tag fg20070202-rewind and look at $MARLIN/src/Testprocessor.cc.

Dear Frank, sorry but I'm not able to find the tag you mentioned neither from the web interface (http://www-zeuthen.desy.de/lc-cgi-bin/cvsweb.cgi/) nor accessing the repository directly:

Quote:
[toto@localhost ~]$ export CVS_RSH=ccvssh
[toto@localhost ~]$ export CVSROOT=:ext:anonymous@cvssrv.ifh.de:/marlin
[toto@localhost ~]$ ccvssh login

Enter password:
Login ok.
[toto@localhost ~]$ cvs co -r fg20070202-rewind Marlin
cvs [checkout aborted]: no such tag fg20070202-rewind

I believe that your modifications are included into the main branch.

Cheers, toto

Subject: Re: A Marlin processor for pedestal calculation
Posted by gaede on Mon, 05 Feb 2007 16:25:48 GMT
View Forum Message <> Reply to Message

Hi Antonio,

sorry, sth. must have gone wrong with the tag - it exists now - both on the web-interface and of course in the repository. Please try again.

 Cheers, Frank.

PS: If you checked out the current HEAD the code is identical...

Subject: Re: A Marlin processor for pedestal calculation
Posted by antonio.bulgheroni on Wed, 07 Feb 2007 11:36:15 GMT
View Forum Message <> Reply to Message

Dear Frank and other,
     I tried the Rewind feature you added and I believe we are very close to the solution. The point is the following, your example is perfectly working when I want to re-loop on a subset of events. This may happen but, at least for pedestal, it is also very common to loop over one full file. I browsed your modifications especially in the Marlin.cc file and I discovered that Rewind exceptions are caught only from the processEvent(LCEvent *) and not from end(). Do you think it is possible to add a try/catch block also there?

This will be perfect in both cases, if one wants to calculate pedestal just on a certain event range then for all other events the processEvent(LCEvent*) method is returning immediately and the rewind (if needed) is done in the end() method.

If this is not possible, then I would like to know how many events are saved into the file. So that in the processEvent(LCEvent*) I can add a isLastEvent() block. Is this information available somewhere?

Another question, again related with the end() and/or isLastEvent(). The main outputs of the PedestalNoiseProcessor are just three vectors: one for pedestal, one for noise and the last one for the pixel status. These three vectors are continuously updated during the processing, but their final value is available only at the end of the last loop, so in the end() method. The original idea was to save in the output LCIO file just a collection with three TrackerData objects (a single collection in a single event). The point is that the end() method doesn't have access anymore to the LCEvent. I should do it in the processEvent(LCEvent*) but only for the last event.

Another possibility is to store the three vectors into a "Condition" file, but I still have to better investigate this opportunity.

What do you think about?

Thanks in advance for your reply...

Cheers, Antonio

Subject: Re: A Marlin processor for pedestal calculation
Posted by gaede on Fri, 09 Feb 2007 10:46:01 GMT
View Forum Message <> Reply to Message

Hi Antonio,

I am glad that the 'rewind feature' works for the pedestal calibration, at least for the best part of it. As pointed out this is somewhat experimental software that I quickly implemented. For a release I would like to iterate on the design and understand the implications and use cases better. For example I think an additional callback 'calibrateEvent(LCEvent* evt)' would make the usage a lot simpler and clearer.

Putting the rewind feature also to the end() method would cause potentially severe problems with resource allocation. The modular design of Marlin provides the init() and end() callbacks to the processors for resource allocation and it is guaranteed that both methods are called exactly once before and after the processing of events respectively.
Also there is currently no way to tell how many events are going to be in the input data stream that is going to be processed (other then reading all event headers in the stream).

I think both things are not really needed for a calibration. One should have a well defined criterion to end the calibration loop and rewind, e.g. the first N events or until all errors of constants are below a certain level.

However if you really want to run the calibration on all events
you can easily do this right now by simply specifying the input stream twice and toggling the calibration flag if the first event is presented for the second time.

If you want to make your calibration persistent I would suggest to use LCCD the conditions data toolkit. With this you can for example store the calibration in a plain LCIO file and then easily access the calibration in a Marlin job that processes the events.
LCCD has various options on how the conditions (calibration) data are stored/read: database, plain LCIO file, LCIO file with DB dump or in the Data file directly. Please check the documentation and examples.

 Cheers, Frank.

Subject: Re: RunHeader EUDET specific parameters
Posted by antonio.bulgheroni on Fri, 09 Feb 2007 13:28:33 GMT
View Forum Message <> Reply to Message

I think that we also need to add an integer parameter with the number of events in the file.

In fact, since it is not possible to dig out this information from the file and since there are some processes (like pedestal, eta function calculation ...) that require multiple loops on events it become of vital importance to know this number in advance...

 NoOfEvent (int)

Cheers,

Subject: Important steps toward the EUDET telescope analysis
Posted by antonio.bulgheroni on Mon, 19 Feb 2007 14:17:55 GMT
View Forum Message <> Reply to Message

Dear All!

   This short post to announce that we are very close to the first release of the EUTelescope
Marlin processors. The code comes with doxygen documentation, but since we still don't have a
place to publish it, I will try to describe here what has been already implemented and what is still
missing. For the time being I tried to avoid ROOT linkage and I'm using AIDA for histogramming.

This is what we already have:

 EUTelSucimaImagerReader. This is just a test to implement the on-line conversion of the
standard SUCIMA DAQ format. Marlin reads an event in the SUCIMA DAQ format and converts it
on the fly into a LCEvent. In principle we can implement a reader also for the IRES USB Imager
board.
 EUTelPedestalNoiseProcessor. This processor calculates the pedestal and the noise value of
each pixel of each detector in the telescope. Few algorithm have been implemented and waiting
to be fully debugged. The results of this processor are saved into a "Condition" file (it can also be
a database entry, but not sure it is really needed).
 EUTelCalibrateEventProcessor. This processor removes the pedestal from the current rawdata
and eventually suppress the common mode. The output of this processor is a collection of
TrackerData object, one per detector, with the data ready for the cluster search.
 EUTelClusteringProcessor. This processor looks into each detector TrackerData for group of
pixels passing the cluster selection cuts. The output of this processor is a collection of
TrackerData containing only the clusters.

What is missing:

 EUTelAutoPedestalProcessor. This processor will be used with self-biased pixel detectors in
which there is actually no need for a real pedestal calculation. The pedestal and noise values for
each pixel can be initialize to a reasonable value and then, during the first N events updates to the
correct value.
 EUTelClusterMergingProcessor. This processor will calculate the distance between clusters
belonging to the same detector. In the case the two clusters are touching each other, something
should be done to separate them. Probably not really needed in a telescope setup where the
number of clusters per plain should be sufficiently low.
 EUTelDataQualityMonitor. This processor should take as input the clusters provided by the
EUTelClusteringProcessor and fill in sets of histograms/profiles/distributions... to cross-check the
data quality at the detector level. Typical examples are the SNR for each detector, the beam

profile, the eta function...
 All the remaining track finding / fitting stuff.
 Interfacing to the geometry database.

You can download the source code from here or via anonymous CVS. Once you have the source code, uncompress it into $(MARLIN)/packages and type make from the Marlin top folder.

Type make doc to produce the documentation.

To produce some example files browse to the $(MARLIN)/packages/Eutelescope/test and read the README files.

Please give it a try and let me known what you think about. Your feedback is very well appreciated!

Best regards,

Antonio

Subject: Re: A Marlin processor for pedestal calculation
Posted by emlyncorrin on Fri, 02 Mar 2007 12:46:37 GMT
View Forum Message <> Reply to Message

I have another idea of a possible way to handle this. In the Bonn/Mannheim DAQ, they have a beginning-of-run-event (BORE) and an end-of-run-event (EORE) that were sent before the first and after the last real events in a run. They were basically a way for the individual producers to send header/trailer information to the DAQ.
I like the idea of this, and I am going to implement something similar in the final DAQ. These events don't have to be saved to the LCIO output file, but if they are, then the EORE could be used to detect the end of a run, before reaching the end of the file, and without needing to put the number of events in the header.

Subject: Re: A Marlin processor for pedestal calculation
Posted by antonio.bulgheroni on Fri, 02 Mar 2007 14:16:49 GMT

Dear Emlyn,
 I like this option too, we probably just need to implement a LCGenericObject with the BORE / EORE information.

Cheers,

Antonio

Subject: ilcinstall: configuration file for Eutelescope
Posted by antonio.bulgheroni on Mon, 14 May 2007 10:08:40 GMT

Dear all,
    this is just to inform you that, thank to the new ilcinstall script, it is possible to install all the ILC software tools in a single process including additional Marlin packages as Eutelescope.

In the attached file, the configuration script I used to install all the ILC software together with the Eutelescope package.

Cheers,

Antonio

```
File Attachments
```
1) install.cfg, downloaded 1533 times

Subject: EORE file fix
Posted by antonio.bulgheroni on Wed, 23 May 2007 11:54:07 GMT

Dear all,
    during the last meeting in Geneva we concluded that we need something like a BORE/EORE identification. In case the acquisition software was hanging we need to have a sort of "file fix" utility able to scan the LCIO file and eventually append an EORE in the end.

I prepared this utility as a Marlin processor (see details below).

Another important use of this utility is more related to the analysis itself (this is actually the reason why I prepared it).

The point is the following: since Marlin is "Modular" and we are preparing separate processors for each task, one can imagine to perform the analysis step by step keeping track of all the intermediate output files. Consequently every intermediate files should have the same structure (say with an EORE at the end). This is trivial whenever all records within the input file are processed, but if the user limits the number of record to something below the total number, then the output file won't have the EORE.

In Marlin framework, a workaround can be easily applied implementing an EUTelOutputProcessor inheriting from the standard LCIOOutputProcessor. When the EUTelOutputProcessor::end() is called by the ProcessorMgr, a check on the type of the previous event is done. If this is an EORE then just close the file, if not append an EORE before closing.

From an OO point of view, this just requires to re-implement the end() method keeping all the others as in LCIOOutputProcessor. It is very easy and I've done it already.

From a technical point of view, there a small detail to be fixed in a the original LCIOOutputProcessor implementation. Since I want to derive EUTelOutputProcessor from LCIOOutputProcessor and not from Processor directly, I (think I) need to have also the LCIOOutputProcessor(const std::string& typeName) constructor as it is in the DataSourceProcessor.

This modification along with another change to make the new messaging system working also with const method is included in the attached patch. Frank, can you comment on that? Do you think it is worth or we are moving in the wrong direction?

Thanks,

toto

File Attachments
1) marlin-v00-09-07-p01.patch, downloaded 1270 times

Subject: Documentation server
Posted by antonio.bulgheroni on Thu, 31 May 2007 15:03:17 GMT
View Forum Message <> Reply to Message

Dear all!

  These days I made a major update of the EUTelescope documentation server. I added also

some introductory pages explaining what it is used for and how an end user should use it.

Everything is available here

 http://www.roma3.infn.it/~bulgheroni/Eutelescope/head/index. html

Please consider visiting it!

Cheers,

Subject: TB1 data taking phase over
Posted by antonio.bulgheroni on Wed, 20 Jun 2007 12:36:50 GMT
View Forum Message <> Reply to Message

Dear all,
 this is to inform all of you that our first test beam is now over, at least for what data taking is
concerning.

 We were able to run the brand new analysis software fully compliant with the ILC standard tools
and we are now producing the first very fresh results. Few of them have been presented already
this morning in DESY during the summary meeting (agenda & talk available at indico server).

Now data are migrating to the GRID for storage and continue the analysis also from outside
DESY.

As soon as they will find their final location, I will post here a small how to access to them and a
description of the dataset catalog.

Thanks to everybody for the very fruitful collaboration and stay tuned to see more.

Cheers,

Antonio on behalf of the testbeam24 crew!

Subject: Re: Documentation server
Posted by iglesias on Tue, 20 Oct 2009 14:17:27 GMT
View Forum Message <> Reply to Message

Hello:

I would like to use the EUTelescope package to analyse the TestBeam data from DEPFET, I try to open the web where you put all the documentation but there are problems with the link.

  http://www.roma3.infn.it/~bulgheroni/Eutelescope/head/index. html

Regards

Carmen

Subject: Re: Documentation server
Posted by rubinsky on Wed, 21 Oct 2009 06:38:00 GMT
View Forum Message <> Reply to Message

Dear Carmen,

the documentation moved (very recently) to

http://projects.hepforge.org/eudaq/Eutelescope/

cheers
Igor

iglesias wrote on Tue, 20 October 2009 07:17Hello:
I would like to use the EUTelescope package to analyse the TestBeam data from DEPFET, I try to open the web where you put all the documentation but there are problems with the link.

  http://www.roma3.infn.it/~bulgheroni/Eutelescope/head/index. html

Regards

Carmen

Subject: ilcsoft v01-14 error in pedestalmerge compilation

Hello,

I'm trying to build the ilcsoft version 01-14, all compile fine until it arrives at the end of the
EUTELESCOPE compilation. It breaks in the pedestalmerge file.

[100%] Building CXX object CMakeFiles/pedestalmerge.dir/src/exec/pedestalmerge.cxx.o
 /data/lhcb09/a/jtrenado/ilc/v01-14-01/Eutelescope/v00-06-03/ src/exec/pedestalmerge.cxx: In
function 'int main(int, char**)':
 /data/lhcb09/a/jtrenado/ilc/v01-14-01/Eutelescope/v00-06-03/ src/exec/pedestalmerge.cxx:169:
error: 'TrackerRawDataImpl' no se declaró en este ámbito
 /data/lhcb09/a/jtrenado/ilc/v01-14-01/Eutelescope/v00-06-03/ src/exec/pedestalmerge.cxx:169:
error: el argumento de plantilla 1 es inválido
 /data/lhcb09/a/jtrenado/ilc/v01-14-01/Eutelescope/v00-06-03/ src/exec/pedestalmerge.cxx:169:
error: invalid type in declaration before '(' token
 /data/lhcb09/a/jtrenado/ilc/v01-14-01/Eutelescope/v00-06-03/ src/exec/pedestalmerge.cxx:169:
error: se trata la lista de expresiones initializer como una expresión compuesta
 /data/lhcb09/a/jtrenado/ilc/v01-14-01/Eutelescope/v00-06-03/ src/exec/pedestalmerge.cxx:169:
aviso: left-hand operand of comma no tiene efecto
 /data/lhcb09/a/jtrenado/ilc/v01-14-01/Eutelescope/v00-06-03/ src/exec/pedestalmerge.cxx:169:
error: conversión inválida de 'IMPL::LCCollectionVec*' a 'int'
 /data/lhcb09/a/jtrenado/ilc/v01-14-01/Eutelescope/v00-06-03/ src/exec/pedestalmerge.cxx:169:
aviso: variable 'encoderEncoder' sin usar
 /data/lhcb09/a/jtrenado/ilc/v01-14-01/Eutelescope/v00-06-03/ src/exec/pedestalmerge.cxx:171:
error: 'TrackerDataImpl' no se declaró en este ámbito
 /data/lhcb09/a/jtrenado/ilc/v01-14-01/Eutelescope/v00-06-03/ src/exec/pedestalmerge.cxx:171:
error: el argumento de plantilla 1 es inválido
 /data/lhcb09/a/jtrenado/ilc/v01-14-01/Eutelescope/v00-06-03/ src/exec/pedestalmerge.cxx:171:
error: invalid type in declaration before '(' token
 /data/lhcb09/a/jtrenado/ilc/v01-14-01/Eutelescope/v00-06-03/ src/exec/pedestalmerge.cxx:171:
error: se trata la lista de expresiones initializer como una expresión compuesta
 /data/lhcb09/a/jtrenado/ilc/v01-14-01/Eutelescope/v00-06-03/ src/exec/pedestalmerge.cxx:171:
aviso: left-hand operand of comma no tiene efecto
 /data/lhcb09/a/jtrenado/ilc/v01-14-01/Eutelescope/v00-06-03/ src/exec/pedestalmerge.cxx:171:
error: conversión inválida de 'IMPL::LCCollectionVec*' a 'int'
 /data/lhcb09/a/jtrenado/ilc/v01-14-01/Eutelescope/v00-06-03/ src/exec/pedestalmerge.cxx:171:
aviso: variable 'encoderEncoder' sin usar
make[2]: *** [CMakeFiles/pedestalmerge.dir/src/exec/pedestalmerge.cxx.o] Error 1
make[1]: *** [CMakeFiles/pedestalmerge.dir/all] Error 2
make: *** [all] Error 2

Any idea?
Than you very much

Subject: Re: ilcsoft v01-14 error in pedestalmerge compilation
Posted by sailer on Wed, 31 Oct 2012 08:48:40 GMT
View Forum Message <> Reply to Message

Hi Juan,

Do you need the Eudet Telescope software?

If not, you can simply disable its compilation.
If yes, we have to figure out what went wrong...

EDIT:
So I haven't compiled the EUTelescope software in ages (if ever).
But trying it now I got the same error as you.

Apparently some types changed their namespace/heeader files had using statements removed, I
don't know.
But this can be easily fixed by adding "lcio::" in the CellIDDecoder<> in lines 169 and 171
From this

168 if (type == lcio::LCIO::TRACKERRAWDATA) {
169 lcio::CellIDEncoder<TrackerRawDataImpl> encoderEncoder(
eutelescope::EUTELESCOPE::MATRIXDEFAULTENCODING, collectionMap[name]);
170 } else if (type == lcio::LCIO::TRACKERDATA) {
171 lcio::CellIDEncoder<TrackerDataImpl> encoderEncoder(
eutelescope::EUTELESCOPE::MATRIXDEFAULTENCODING, collectionMap[name]);
172 }

to this

168 if (type == lcio::LCIO::TRACKERRAWDATA) {
169 lcio::CellIDEncoder<lcio::TrackerRawDataImpl> encoderEncoder(
eutelescope::EUTELESCOPE::MATRIXDEFAULTENCODING, collectionMap[name]);
170 } else if (type == lcio::LCIO::TRACKERDATA) {
171 lcio::CellIDEncoder<lcio::TrackerDataImpl> encoderEncoder(
eutelescope::EUTELESCOPE::MATRIXDEFAULTENCODING, collectionMap[name]);

172 }

Cheers,
Andre

Subject: Re: ilcsoft v01-14 error in pedestalmerge compilation
Posted by juantrenado on Wed, 31 Oct 2012 12:22:00 GMT
View Forum Message <> Reply to Message

Hi Andre,

than you very much.
It resolves the problem.

Cheers,
 Juan