Subject: FairMCPoint
Posted by Volker Friese on Tue, 06 Mar 2012 11:14:52 GMT
View Forum Message <> Reply to Message

I would like to trigger a discussion of the data classes implemented on the fairroot level, and start off by the class FairMCPoint.

This describes the geometrical intersection of a MCTrack with some sensitive volume. Quite obviously, it must contain three coordinates (floats) and a reference to the MCTrack (integer). In addition, one might want to store the momentum components of the track at this point - three more floats. It makes also sense to have some energy loss and track length. With that, one ends up with 8 floats and one integer - 68 bytes in total.

The size of the current FairMCPoint, however, is 224 bytes, which means that the framework introduces an overhead of some 230%. This is mainly due to the inheritance caused by FairLink.
Objects of the class FairMCPoints are very frequent, and the use of FairMCPoint is obligatory.

The facts above bring me to the opinion that the current implementation is not well architectured. I would thus like to bring forward some theses for the discussion:

The data members of the base class FairMCPoint are legacy from the times when fairroot/cbmroot was developed for a single experiment.
FairMCPoint should not have any data member. Functionality needed for the framework (e.g., event display) should be implemented by abstract accessors.
The application (experiment) should decide which data members to implement in the concrete class, and in which representation (e.g. double, double32, integer).
This, by the way, holds for all data base classes (e.g., MCEventHeader, MCTrack, Hit, Digi etc.).
Making FairMCPoint derive from FairTimeStamp is counterintuitive and not a good design. A 3-d space point is not a special type of a time stamp.
Time stamps on the MC level do not make sense.
While FairLinks is a very useful scheme, the decision whether to use it for a particular data class should be left to the application, which can include the functionality either by making FairLink a member of its data classes, or by multiple inheritance.

Subject: Re: FairMCPoint
Posted by Mohammad Al-Turany on Thu, 08 Mar 2012 16:50:34 GMT
View Forum Message <> Reply to Message

Hallo Volker,

Some of the changes you suggest are straight forward, and as you said they came through the

fact that it was for one experiment, other things like the time stamp came from the time where we did the time stamp in the MC, which we do not do any more. So we can change it easily but this will of course break the compatibility to the old stuff. The rest is more complicated and we need to discuss it first internally then maybe we can organize a meeting with you and other interested users to discuss the details.

regards,

Mohammad

Subject: Re: FairMCPoint
Posted by Mohammad Al-Turany on Wed, 14 Mar 2012 11:55:54 GMT
View Forum Message <> Reply to Message

Hallo Volker,

looking at the current implementation one can improve it as following:
1. The FairBasePoint and FarMCPoint will be unified in one class (FairMCPoint)
2. FairSingleLinkedData is not needed (will be removed)
3. FairLinkedData and FairLinkedMultiData will be unified in one class (FairLinkData)
4. The new FairMCPoint class will inherits directly from FairMultiLinkedData
5. The FairMCPoint will not inherit from FairTimeStamp

The new design will be:

Now the new FairMCPoint and FairHit has to be understood as an example, in fact even now before changing anything you can skip them and define your own classes the only thing the framework need is that you inherits from TObject in ROOT nothing more. Now with the new changes it get clearer and in case you need to use the fair links you inherits from FairlinkData, otherwise you just define your own classes that are TObjects. The same is true for the Hit Class if you went to use our scheme for time base simulation you inherits from the FairTimeStamp, otherwise you define your own.

For the event display using the example point and Hit you can directly use our implementation, or you do it on your own.

With these changes, the code get clearer but the current functionality is not really changed and every body is free to choose where to plug his data into the framework what to re-use, etc.

best regards.

Mohammad

File Attachments

1) fair_points.png, downloaded 334 times

Subject: Re: FairMCPoint
Posted by Volker Friese on Wed, 14 Mar 2012 18:17:46 GMT
View Forum Message <> Reply to Message

Dear Mohammad,

thanks a lot for your proposal. It seems to be a misunderstanding of mine that the base data classes are to be used (inherited from). In fact, if they are to be taken as examples, and not requested by the framework at all, there is of course no need to change them at all.

To your considerations:

Quote:1. The FairBasePoint and FarMCPoint will be unified in one class (FairMCPoint)
Since I do not know the motivation of the class FairBasePoint, I cannot comment.

Quote:2. FairSingleLinkedData is not needed (will be removed)
3. FairLinkedData and FairLinkedMultiData will be unified in one class (FairLinkData)
4. The new FairMCPoint class will inherits directly from FairMultiLinkedData
5. The FairMCPoint will not inherit from FairTimeStamp

I think that all these items would be improvements.

Quote:With these changes, the code get clearer but the current functionality is not really changed and every body is free to choose where to plug his data into the framework what to re-use, etc.
Fine with me.

Best regards,

Volker