Subject: Re: Strategy discussion on Track objects
Posted by Sebastian Neubert on Fri, 08 May 2009 17:41:00 GMT
View Forum Message <> Reply to Message

Dear colleagues,

as far as I understand from your discussion there are several interests that should be fulfilled (and where I have the feeling everybody agrees):

1) There should be a central PndTrack object which is part of PANDAROOT and provides a unified interface between the track-reconstruction and the particle data classes which will be used in analysis code.

2) In order to do fitting and refitting we need a TrackPropagator. The standard solution here is GEANE but it might be interesting to be able to exchange that propagator. There needs to be an interface to pass information forth and back between PndTrack and the propagators.

3) We need fitting routines which have access to residual information between hits and tracks. We need to be able to transfer hit-data and track-data (and track propagation!) from and to the fitting routines - this means the data has to be put into a form that is understood by the fitting framework.

GENFIT provides solutions for point 2) and 3) namely the AbsTrackRep and AbsRecoHit interface classes and the Track class.

Now from my perspective the real conflict in the different possible solutions here is the following:

Should PANDAROOT use the fitting packages (GENFIT) or should GENFIT use the PANDAROOT structures?

I understand that Mohammad defends the latter position as this guarantees that condition number one will be fulfilled, namely a central, unified access point for track data. Furthermore in that way PANDAROOT would not become too dependent on GENFIT.

For GENFIT we were forced to develop the data structure as it is now in order to provide its flexible functionality. These structures are deeply self-consistent which is why Christian is so reluctant to change it (and I strongly support him in this point).

In order to unify all this I would propose the following:

A "translator" has to be written which translates PndTrack into GENFIT language (Track) and the other way round. This would allow us to use the full GENFIT functionality while keeping mutual dependencies minimal (as required in the Computing Model). In software design pattern language this is an "Adapter".
Christian already has agreed to take over what is necessary from the GENFIT side to do this. It would be PANDAROOT responsibility to provide all information GENFIT needs to do its job.

The following scenario would be ideal for me:
One would use the Adapter to initialize GENFIT in the reconstruction, do the Kalman Fit and

use the Adapter again to convert to PndTrack and store this object.

Then you take home your root file. In your analysis script you use a different Adapter to initialize the YAFF* and do a vertex-refit and convert back to PndTrack (+ related objects). You store the result.

Your colleague takes the refitted tree and uses the GENFIT Adapter to perform an extrapolation from the vertex to a certain detector with GEANE.

Best Regards, Sebastian.

*) Yet Another Fitting Framework
