Discussion on Physics Event Model - November 21, 2001

Present:
O.Callot, M.Cattaneo, Ph.Charpentier, G.Corti, M.Frank, P.Mato

Philippe reported about the discussion he had with Daniel Wicke (IDEA) regarding some of the choices Delphi made. A summary of his discussion is available on the web attached to this meeting. Philippe mentioned that it was stressed that each class should contain only the relevant information and nothing more, hence the IDEA choice of a base class and external navigation tree. All the specialized info is in the derived classes that are what users deal with.

The attractiveness of having a base class is so that analysis tools will deal with only the base class. This implies that the base class should provide all possible methods and those relevant should be implemented in the derived classes. Another option is to have a single class where at creation the default values for instance as pointer to reconstructed info is set to zero. Neither sounds too attractive.

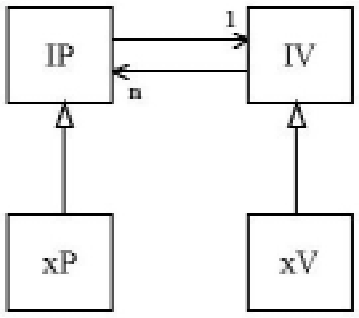
In fact the commonality all particles (generator, simulation, physics) have are essentially a particleID and the fourMomentum, plus being in a "tree" relationship with vertices.

What we want is really to have separate classes where the common thing have the same terminology. Should this be enforced by implementing a common class?

The requirement of doing all operations in all tools is in conflict with implementing the classes. How many tools are really common to all particle categories.
While it would be nice to be able to do vertexing on pure simulated particles (with no smearing), is that really necessary? Beside the navigation issues are other tools really common? If we want to do the

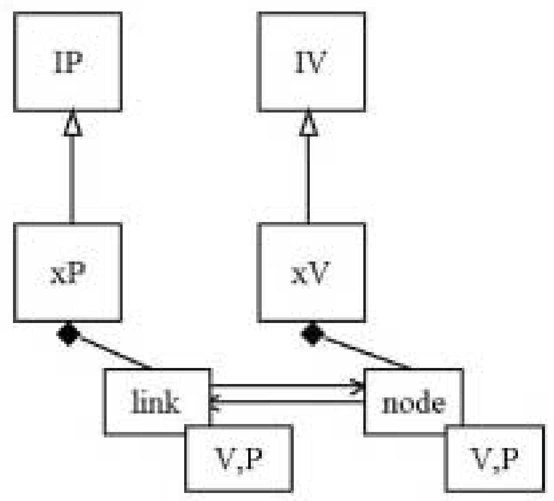
The real common aspect is the navigation of the classes when in trees.

This cannot really be put in the base class because of the language and would imply that the user would have to do dynamic casting, something we should try to avoid.



```
P = getDecay(…); xP = getXDecay(…)
```

Pere suggested having the relationship as a template tree but this brings us back to having an external tree.



`virtual methods in IP, IV`

The relationships cannot be in the base class: but then the different particles category classes are quite different and the relationships is external.

Users should use derived classes where all the “extra” specific info is stored.
Only common tools will get the interface, other tools will be specific to the derived classes.