Points of convention

- All points are only for discussion; we do not necessarily need a convention for each of these. Some points are rather trivial. In roughly descending order of importance:

- How much code shall we plan to inherit from BaBar? BtaCandidates, for example, are not really BaBar-specific, and could be reused. (Not very lightweight, though.) The Beta framework is a bit cluttered with old code, but it has the advantage that it works, and many people are used to it. On the other hand, reinventing circular objects used in transportation is fun!

- Track changes within a detector: If a particle decays in the DIRC, how should we let the EMC (or the user who called the DIRC) know? (Possibly a tracking responsibility; but the probability might change if you change the DIRC model, for example.)

- Detector models: Should we agree on a common method of creating these, and perhaps a common way of calling them? For example:

```
AbsDetectorModel*   AbsDetectorModel::getModel (std::string modelName);
AbsDetectorResponse AbsDetectorModel::getResponse (const Hep3Vector& mom, PdtPid::PidType part, int charge) const;
```

  If all subdetectors have the same methods, life is easier for the user, but may be more difficult for the developer.

- Detector response: Is it possible / desirable to have a unified object for this? *E.g.* (number of SVT hits) == (number of DIRC photons)? Or should each detector have its own struct?

- Lists: STL now has std::vector, std::list and std::map, which may be a better alternative to HepAList.

- Particle identification: `LundID`, `PdtPid::PidType`, other?

- Passing vectors: `Hep3Vector`, (p, θ, ϕ), (p_x, p_y, p_z)?

- Units: Probably best to stick to BaBar convention.
- Strings: CLHEP `HepString`, ROOT `TString`, or stdlib `std::string`?
- Coding standard: One True Brace Style? BaBar style? Other heresy?