Table of Contents

LHCb Physics Event Model Task Force Page

Members

- Vanya Belyaev
- Marco Cattaneo
- Gloria Corti (Event Model Review)
- Ulrik Egede
- Chris Jones
- Patrick Koppenburg (chair)
- Gerhard Raven (Event Model Review)
- Patrick Robbe

Meetings

- First meeting on Tuesday July 5 at 10AM in room 14-4-030. VRVS: Mars.
- Second meeting on Tuesday July 12 at 14AM in room 32-1-014. VRVS: Snow.
- Third meeting on Tuesday July 19 at 10AM in room 14-4-030. VRVS: Neptune.
- Fourth meeting on Tuesday July 26 at 10AM in room 14-4-030. VRVS: Sun.
- PK away on Tuesday August 2. Meeting booked in room 14-4-030. VRVS: Plane.
- Tuesday August 9 at 10AM in room 14-4-030. VRVS: Ocean.
- Tuesday August 16 at 10AM in room 14-4-030. VRVS: Car.
- Tuesday August 23 at 10AM in room 14-4-030. VRVS: Sun.
- Tuesday August 30 at 10AM in room 14-4-030. VRVS: Fog. - **Cancelled**.
- Tuesday September 6 at 10AM in room 14-4-030. VRVS: Boat. - **Cancelled**.
- Tuesday September 20 at 10AM in room 4-S-013. VRVS: Twister.

Remit by Nick Brook

- provide & review use cases & requirements for the (physics) event model
- define "physics" objects for analysis
- ensure the event model will meet future analysis requirements, including access to the truth information
- deadline: 31st July 2005

Scope

Here is a first brain dump of the "scope" by PK

- The essential ingredients are of course the Particle and Vertex classes, as well as their MC-true counterparts (that affect all software, not only DaVinci), but it would be nice if we could also look at the other "physics" classes like the new FlavourTag class for instance.
- The MC part is more urgent as we need to start production early 2006. For the pure DaVinci stuff, more iterations are allowed.
- Now that the new Track class is available a change of the ProtoParticle is mandatory. We'll have to revise how we link to PID and how of much of it we store in the ProtoParticle itself (or if we can get rid of it).
    - (MC) Perhaps we should also define the DST content at the same time as re-defining the ProtoParticle.
- How do we relate a Particle to the PV that has been used to select it (if any)?
- One question we'll have to have in mind is how these objects get persistified and what is the minimal

amount of data one needs to make a physics analysis (please don't answer: "the raw buffer").
- Changes in the event classes cannot be decoupled from changes in the interfaces for the basic tools. For instance the vertex fitters interface depends on how the Particle and the Vertex are connected. We might hence also have to have a look at the most basic tools: the VertexFitter, PhysDesktop, ParticleMaker, FilterCriterion...
- This will bring to the question of how the core DaVinci tools and LoKi should be related... I don't know if we'll have time to address that.

Additional comments

- (VB) Do we want to have the same physics event model for "off-line" and HLT? (personally i hope that the answer is "yes")

General Requirements

- Should we go for the new MathCore library?
    - ♦ yes for vectors,
    - ♦ also for linear algebra. But we don't want them to inherit from TObject.
        - ◊ Require that from the architects forum
    - ♦ There are many problems with these classes. Check out the MathCore wiki page.
- A general comment is that whenever there is a convention, it should be agreed and very well documented
- We would also like to have something like "unix-like" links in the TES. One should be able to write a subset of a set of objects into some location without having to clone the objects
- There are many cases below where one would like to associate an object and some property, to be put on the TES. There should be a generic way of doing that.

ExtraInformation

- There are many places (see below) where we would like to be able to store some additional "user" information in a class. This could be coded in a new ExtraInformation class that would essentially contain a std::vector< std::pair<int,double> > where the int would come form an enum or set by the user (by some convention that numbers above N are set by users).
    - ♦ Vanya will provide an implementation in a few days.
    - ♦ Done: see doxygen

There is some discussion about the complicated structure of the class.

Analysis classes requirements

Particle

- (LHCb Doxygen) - (Proposal Doxygen)

The particle is the object the user deals with in a physics analysis. It is created from *reconstructed* objects, where this usually means ProtoParticle.

- A Particle corresponds as much as possible to the intuitive understanding of a particle
- It has a defined PID
- It allows to easily access:
    - ♦ the charge,
    - ♦ the PID,
    - ♦ the mass,

        - ◊ There are several ways of getting the "mass" of a particle. One should make clear that the number returned by this method is the measured mass (from the result of the vertex fit?). -> Rename the mass() method to measuredMass()
    - the 4-momentum,
    - "a" position (creation vertex or first measurement),
    - the error matrix
    - the decay products
- It should be possible to navigate back to the mother if available
- Association to MC truth should be kept by default after the particle is cloned.
- The user should not have to access the underlying objects in a typical analysis
- But, one should be able to navigate back to the *reconstructed* objects it is based on.
- Final and intermediate states have to be treated the same way
- Charged and neutrals have to be treated the same way
    - This implies some thinking about how to deal with photons (or K_L, but we usually don't care) and pi0s
- It should not depend on MC truth
- One should be able to relate a Particle to a Primary Vertex
    - This could be saved in some table relating Particle and PV with some weight, typically IP/error.
    - The first step of an analysis would then be to select Particles according to this weight

Particle Proposal

- (Proposal Doxygen)

A crazy scenario has been considered: Particle would be essentially an interface to for instance ParticleFromTrack, ParticleFromNeutral, CompositeParticle. Users would always get Particles from the PhysDesktop but always create CompositeParticles. This would allow to save some space as some internal variables don't always have a meaning, but it makes the model less intuitive.

We are more in favour of a minor modification of the track

- Rename the mass() method to measuredMass()
- The decay tree is stored in the Particle. The coherence of the particle tree and the vertex tree has to be ensured.
- All primary "stable" reconstructed particles are made from ProtoParticle. Remove Particle making from TrgTrack and the classes TrgCaloParticle, CaloParticle...
    - One should have a pointer to ProtoParticle available, sometimes 0.
    - The assumption that a Particle with no origin is a composite becomes wrong in the case of Particles from MCParticle. -> Provide a method bool isBasicParticle(){return m_daughters.empty();}.
    - In the case of a Particle made from MCParticle, the navigation is done via MC associators, like for reconstructed objects
- ParticleVector should be std::vector<const Particle*> instead of std::vector<Particle*>. Do we need both?
    - To be discussed with Stefan
    - We prefer Particle::Vector (and ::ConstVector) instead of ParticleVector. It allows to get the iterator in a templated method.
- Add a "user flag" allowing to figure out how the particle has been built
    - This could be stored in the suggested ExtraInformation class
- Remove slopeX(), slopeY()
- m_pointOnTrack should become m_referencePoint
- isResonance() is useless. This should be a property of the vertex fitters.

July 26 modifications:

- Added an enum of additional information. Typical fields are "weight" and "ConfLevel"
- Added methods wieght() and confLevel() that read these fields in the ExtraInfo.
    - ♦ replaces the confLevel attribute
- Removed Key and Info typedefs as the ExtraInfo typedef defines that ket has to be int and info has to be double. This reduces the confusion a little bit. Guess that Vanya will disagree ;-).
- Merged all covariance matrices into one big 7x7 HepSymMatrix. Ther are methods to access the sub-matrices.
    - ♦ The slope errors get lost
    - ♦ Comment by Vanya: If CLHEP(or whatever matrix library) does not allow to address submatrix through reference, we should not store the whole 7x7 matrix.. In this case the storage by submatrices is more efficient, and as for "whole" 7x7 matrix one needs to introduce a method. Of course , **IF** Linear ALgebra classes from ROOT, allows to interact with "matrix-views" (submatrices as references) one definitely should use the 7x7 solution.
- Charge is int
- The ExtraInfo attribute is called m_additionalInformation, but nobody is supposed to access it directly. The relevant methods are info, addInfo, eraseInfo, hasInfo.
- Removed the m_desktop attribute. Will try to replace it by a map of <Particle*,bool> in the PhysDesktop.

August 16 modifications:

- Use ROOT matrix. Allows to get all sub-matrices by reference (?)

This reduces the number of attiributes to 10 (from originally 15).

ProtoParticle

- (LHCb Doxygen) - (Proposal Doxygen)

The ProtoParticle links the reconstruction and PID information. It has no defined PID. It is the last step of the reconstruction stage

- It allows to link to all relevant reconstructed and PID objects
- It should allow to access PID likelihoods
- The PID information from the various detectors should be as generic as possible, typically vector of <ID, double>

ProtoParticle proposal

- (Proposal Doxygen)

- Merge m_pIDDetectors and m_pIDinfo, since combining the pid is a technique.
    - ♦ Use ExtraInformation class
    - ♦ One would have a an enum with fields like RICHElectronID, CaloElectronID, GlobalElectronID, even GlobalElectronIDWithoutRICH...
- Do we want to keep all SmartRefs as they are of enforce a SmartRefVector where all PID classes interface PID?
- no method bestPID().
- merge detPid() and detPidValue()

July 26 modifications:

- Same modifications as for Particle wrt the ExtraInformation.
- added TrackChi2, TrackNDof, TrackType to ExtraInformation enum.
- removed pointers to state. State is owned by Track now, there's no other way of getting it whan asking the track for it.

Vertex

- (LHCb Doxygen) - (Proposal Doxygen)

The Vertex class describes a reconstructed decay

- It allows to access
  - ♦ Position and error
  - ♦ chi2
    - ◊ the chi2 should be defined by a technique identified -> add a m_technique variable to be set by the vertex creator
  - ♦ decay products
    - ◊ They would have a different meaning from what is in MCVertex (daughters) and what is in use presently. Hence we suggest to rename the products() method to something different. fittedParticles()? memberParticles() ?
- It should be possible to navigate back to the mother if available. (GR) one could imagine a generic (secondary) vertex searches a la LEP/SLD -- those vertices would not have 'mothers' just decay products...
- The decay products are the particles that have been used in the vertex fit
  - ♦ The coherence of the particle tree and the vertex tree has to be ensured.

Vertex proposal

- (Proposal Doxygen)

July 26 modifications:

- Added user information as for the other classes.
- Renamed positionErr to corrMatrix for consistency with Particle.
  - ♦ Should it be posCorrMatrix for consistency with the method in Particle?
- Daughters are now "outgoingParticles". Suggestions for a shorter name are welcome.

August 16 modifications:

- Use ROOT matrix. Allows to get all sub-matrices by reference (?)

PrimVertex

- (LHCb Doxygen)
- (Proposal Doxygen)

Holds the primary vertices

- The primary vertex is a Vertex.
  - ♦ (GR) What is the 'mother' of a primary vertex?

PrimVertex proposal

- (Proposal Doxygen)

MC truth classes requirements

We have to agree quickly on the *contents* of the classes that would be written out to file. The structure of the class can be revised as long as it is based on the same input.

GenHeader and other headers

- (LHCb Doxygen)

Header for event generator (bunch crossing) information.

- Should contain the event type of *all* collisions
  - -> SmartRefVector < <nopGenCollision >
- Information about the duration and lifetime of a fill also goes here

One would also like to have a **MCHeader** class that contains some general information about how the event has been generated.

- Can this be merged with GenHeader?
- Pointers to the PVs could go in there

Generally we had the idea that one might want to have a summary head class for each processing phase: GenHeader, MCHeader, DigiHeader, RecHeader, PhysHeader...

- These headers could store a vector of randon number seeds so that each step can be redone individually.

ProcessHeader proposal

- (Proposal Doxygen)

Base class for all process header classes. So far there is only GenHeader. We might need MCHeader, DigiHeader...

August 17 Proposal:

- Name and version of the application (in the wide sense, could be pythia)
- vector of random seeds

GenHeader proposal

- (Proposal Doxygen)

August 17 Proposal:

- Added fillDuration and fillLifetime
  - Removed again
- Added pointer to collisions
- Inherits from ProcessHeader

MCHeader proposal

- (Proposal Doxygen)

August 29 Proposal:

- pointer to MC PVs
- Inherits from ProcessHeader

Collision, HardInfo, GenCollision

Collision (LHCb Doxygen)
: Information about pp interaction: signal, PV. Remove PV stuff and put a SmartRef < HepMCEvent > instead.

HardInfo (LHCb Doxygen)
: Holds parameters associated to hard scatter: Mandelstam variables and Bjorken-x for both incoming partons. Filled from Pythia.

GenCollision (LHCb Doxygen)
: Extended generator information for collision. Contains pointer to HardInfo. Derived from Collision.

- All these classes should merge into GenCollision

GenCollision proposal

- (Proposal Doxygen)

August 17 proposal:

- Merger of the three classes.
- Pointer to primary vertex (was not active). Remove position of PV.

August 29:

- Move GeneratorName from HepMCEvent?

MCParticle

- (LHCb Doxygen) - (Proposal Doxygen)

The MC true counterpart of the Particle.

- The look and feel should be similar to the Particle, i.e. similar requirements
- One should be able to perform a fake physics analysis using MC truth, which means one should be able to make Particles from MCParticles
- The method mass() has to be renamed to generatedMass() in order to make clear the meaning and the difference with the measuredMass() in Particle.
    - ♦ That's rubbish. The method mass() does not exist: it's virtualMass(), which is fine.
- The helicity() method is only relevant in the HepMC world -> remove it.
- Remove the hasOscillated() method. This information can go into an external property.
    - ♦ Go to ExtraInformation class?
    - ♦ No: Get it from Vertex type
- There should be a method primaryVertex() that navigates back to the PV of the collision.

MCParticle proposal

- (Proposal Doxygen)

August 17 proposal:

- Use ROOT vectors
- Remove helicity flag
- Remove hasOscillated flag. Provide method instead (see under Vertex)
- New primaryVertex() method that goes up the tree

MCVertex

- (LHCb Doxygen) - (Proposal Doxygen)

MC vertex. Information about type of decay.

- Get rid of collision() member. Use a collision() function instead that allows to navigate back.
    - The association can be done via an exteranl PV-Collision linker class
- Reorder vertex types. Add a OscillatedAndDecay vertex type.

MCVertex proposal

- (Proposal Doxygen)

August 17 proposal:

- Use ROOT vectors
- New primaryVertex() method that goes up the tree
- Add OscillatedAndDecay in enum for oscillation.
    - Rename Decays to DecayVertex to make sure code does not compile anymore
    - Provide isDecay() method instead
- Reshuffled order of enum

MCProperty

- (LHCb Doxygen)

Give an int attribute to an MC particle.

- Associates an int to a MCParticle.
- No changes foreseen

HepMCEvent

- (LHCb Doxygen)

Gaudi wrapper for HepMC

GenMCLink

- (LHCb Doxygen)

Links HepMC and MCParticles.

- Presently being studied by the jets group. Wait for their report.
    - ♦ Conclusion: can be dropped. A relations table is fine.
- Check with tagging people who use it to find the signal B

Related requirements

One should also have a look at the basic tools. We have to make sure that the fucntionality removed from the event model is provided by some helper tool. Here is a list of requirements to the tools using the event model.

- Vertex fitters that modify the input particles sould have a method taht returns a clone of these particles, so that the they can be stored in the TES.
    - ♦ The PhysDesktop already does that automatically
- Once the review is over, Patrick and Vanya have to sit together to review the core DaVinci packages and LoKi. A merger of LoKi::Algo and DVAlgorithm is desirable.
- Comment about photons: The PhotonTool is essentially an extrapolator for photons. It should be used inside the extrapolator, not the user code.
- We would like to be able to write out a "nano-DST" containing only the selected particles but with enough information to redo the fits. For instance the B, the full decay tree and everything needed to make it.
- We would like to be able to save the options that have been modified when creating a DST in the DST itself. This is the only way to ensure that one does not loose the conditions in which it has been created.

Status of Proposal

- Proposal Doxygen

Done:

- 07/25: First versions of Particle, ProtoParticle, Vertex, PrimVertex. MC truth untouched.
    - ♦ Event/PhysEvent tag PEMTF0726 on v9r0b branch
- 07/26: Updated Particle, ProtoParticle and Vertex classes following the discussion on the July 26 meeting.
- 08/10: Updated Particle, ProtoParticle and Vertex classes following the discussion on the August 9 meeting.
- 08/16: Starting to modify MC truth.
- 08/29: More about MC truth. Added a LHCbTypedefs file with some useful typdefs. They are used throughout the event model.
- 09/09: Started implementation of Minimal DaVinci. DaVinciKernel, DaVinciFilter, VertexFitter (minimal version) and DaVinciTools are done. Missing are the more tricky ParticleMaker and GlobalReco
    - ♦ Decided *not* to use MathCore for the moment. See the MathCore wiki page for details.
        - ◊ The CLHEP version is here: Proposal Doxygen

Next steps:

- Implement MC truth
- Compile minimal DaVinci: ChargedProtoPAlg, CombinedParticleMaker, DVAlgorithm, PhysDesktop, Extrapolators, introduce new Vertex fitters.
- Test minimal DaVinci: Build some J/psi.

Trash (of the TF)

V0

- (LHCb Doxygen) To be removed

MCEffTree and EffCheckResult

Not persistified: should not be in Event Model.

Links

- EventModelReview
- LHCbTrackModelTaskForce
- Doxygen of latest LHCb
- MathCore doxygen
    - ♦ MathCore wiki page
- Pages of the 2001 Physics Event Model group
- DaVinci page
- Gauss page
- Proposal Doxygen

-- PatrickKoppenburg - 01 Jul 2005 -- PatrickKoppenburg - 05 Jul 2005 -- PatrickKoppenburg - 13 Jul 2005 -- PatrickKoppenburg - 16 Aug 2005

This topic: LHCb > LHCbPhysicsEventModelTaskForce
Topic revision: r30 - 2006-11-17 - unknown