Moving forward with the overall event model

RD Schaffer

Outline:

- Where are we currently with the event model
- Where do we need to go in the near future

Where are we currently with the event model

- Up to now, the event model (design and implementation) has concentrated on providing access to the raw data (Geant 3 digits)
  - By event model, we mean both persistent and transient event models
  - Access to raw data is needed for any realistic reconstruction work to begin
- We now need to extend the event model to incorporate ALL event objects (I.e. ESD, AOD, tag)
  - There are known scalability problems with the current design

What are the major characteristics of the current event model

- Model allows for the loading of raw data from different sources in a transparent manner (e.g. from Zebra/Objy)
  - Uses strategy objects inside the event structure => e.g. there is no explicit external loading digit sets
- Organization of and access to raw data is done via an identification scheme which follows a logical decomposition of the detector
  - e.g. organised by barrel_endcap/layer/rings/phi sectors
  - simple “collector” objects are used to get digits within an identification range

Where do we need to go in the near future

Anyone interested in working on the overall event design are certainly welcome. For the moment, the interested people are Simona Rolli and myself

Overview of questions to be understood:

- What is role of the event in the software architecture?
- What is the coupling of the database to the application (transient/persistent question)?
- What types of objects are stored in events?
- How are event objects stored and accessed?
- Navigating between event objects?

Where do we go, cont.

The role of the event, some basic assumptions:

- The architecture of our component model has modules which communicate explicitly between themselves:
  - The type of object flowing between modules is specified,
  - There is no need for the Event to serve as the reservoir for objects being communicated between modules
  - One loads ONLY “persistent-capable” objects into an event

  (Note: this does NOT mean that event objects inherit from ooObj.)

Where do we go, cont.

What is the coupling of the database to the application?

In the computing review, it is recommended that we adopt the transient/persistent model

This implies that clients “see” only objects from transient classes

However, this does not answer all questions, for example:

- Write model: What model do we want for deciding what to keep?
  - Load and keep all (at create or load time)
  - Load and select later (explicit decision at later point)

Where do we go, cont.

- Read model: how and when do we decide what to load from the event
  - Are the event elements to be loaded decided at the beginning of the run, or is there some sort of dynamic loading?
  - How does the “navigation” of relationships work with a model of dynamic/non-dynamic loading of event objects
- What is the coupling between persistent and transient objects?
  - Do we always copy?
  - Or is some sort of transient wrapping needed?
  - Or a mixture of both?

Where do we go, cont.

What types of objects are stored in events?

- We need to come up with an abstract interface for “event objects”, e.g. some container-like objects.

How are event objects stored and accessed?

- Are objects accessed via some identification mechanism?
- Do clients navigate within the event, or simply make requests based upon type and/or identification?

Where do we go, cont.

Navigating between event objects?

- How are associations defined between event objects?
- This is coupled to the question of the event object loading.

I believe that a number of the higher level design questions must be addressed in the near term and put into the overall architecture document to be developed in the next few months.

Then prototyping of several of the elements will be needed.