Geant4 Requirements from HEP: Intensity Frontier

Krzysztof Genser, Robert Hatcher/Fermilab

Geant4 Collaboration Meeting, Chartres, France

September, 2012

Two Simulation Domains of neutrino experiments

- Simulation of the neutrino beam lines
  - protons impinging on target at 120 GeV (NuMI)
    - carbon target (or beryllium possibly in the future)
  - protons of 9 GeV on beryllium (Booster)
  - modeling hadron production is critical in both
- Detector simulation
  - generally shower and lepton energies ~0.25-25.0 GeV
    - electron showers are an important NOvA physics signature
      - G4UrbanMscModel93 issue; work around (air ⇒ vacuum)
      - solved yet? proposed 4.9.5 fix wasn't sufficient; 4.9.6?
    - standardized (example) integration of charm/tau decay?
      - feels like each expt has to re-invent/implement common ideas
    - muon-nucleus scattering of interest to some

Comment on Performance

- Efficient use of available resources is always a good thing, but getting it “right” is primary

Mu2e Geant4 Usage

- Mu2e is an experiment looking for neutrino-less muon-to-electron conversion in the Coulomb field of a nucleus (^{27}Al target) currently under design at Fermilab
- Given that the experiment is looking for a very rare process, background simulations are very important
- Mu2e has been using Geant4 v9.4.p02 and is transitioning to v9.5.p01 now
  - QGSP_BERT_HP is the main physics list used now

Summary of Mu2e Geant4 Wishes

1. Refactor G4RunManager::BeamOn
2. Include elements of Kevin Lynch's muon capture (exotic atoms) code in the toolkit, or provide an equivalent functionality
3. Provide a way to access all Geant4 extensions to the PDG particle information independent of the particle creation
4. Indicate when Geant4 does(not) take ownership of pointees

G4RunManager::BeamOn

- Mu2e has chosen to use a non-Geant4 framework to drive our event loop
  - the framework is art, supported by Fermilab Scientific Computing Division.
- Geant4 also wants to own the event loop. Mu2e solution is sketched on the next transparency
  - Mu2e run manager class inherits from G4RunManager and refactors the the BeamOn method into four pieces.

G4RunManager::BeamOn cont'd

```
class Mu2eG4RunManager : public G4RunManager{
  public:
    Mu2eG4RunManager();
    virtual ~Mu2eG4RunManager();
    virtual void BeamOnBeginRun ( unsigned int runNumber,
                                  const char* macroFile=0,
                                  G4int n_select=-1);
    virtual void BeamOnDoOneEvent( int eventNumber );
    virtual void BeamOnEndEvent();
    virtual void BeamOnEndRun();
};
```

G4RunManager::BeamOn cont'd

- These four BeamOn function variants are called from art framework, in the appropriate places. In this way, art framework drives the event loop.
  - The issue is that one needs to potentially maintain this code every time we move to a new Geant4 release.
- It would be a great help to us if the Geant4 collaboration refactored BeamOn so that it is written in terms of a small number of other functions, not necessarily the ones shown earlier.
  - Those who wish to drive their own event loop can call these other functions directly.
  - This would remove the burden of potentially maintaining experiment specific code with every new release of Geant4.
- Mu2e understands that it would require consultation with many experiments to choose the appropriate refactoring.

Include new Muon Capture Code in a Future Geant4 Release

- Mu2e/Fermilab Geant4 group is working with the hadronic group on extending Geant4 to include Kevin Lynch's (now with CUNY/Mu2e) code/ approach to muon capture and muonic (and possibly other exotic) atoms/molecules formation, decays and other related processes (or provide an equivalent functionality).
- To address most of the current Mu2e needs (given that only heavy (~^{27}Al) isotopes are involved) Vladimir Ivanchenko suggested (and had subsequently implemented) a revised code using a new, more modular and generic approach.
- Mu2e appreciates the work and would like it to continue.

Access to Geant4 PDG Particle ID Code Extensions

- Mu2e frequently runs jobs that process events through Geant4 and write these events to disk. Mu2e has a code that reads back these events to do reconstruction and analysis. In the reconstruction and analysis phase, Geant4 is not initialized, therefore the Geant4 particle data table is not available. Instead Mu2e has its own particle data table (a wrapper around HEPPDT).
- Mu2e would like to extend its particle data table to include Geant4 particle data codes beyond those defined by the PDG table.
  - In particular Mu2e would like to include nuclei and ion codes, the corresponding, names, masses, A, Z, electric charge and so on.

Access to Geant4 PDG Particle ID Code Extensions cont'd

- One cannot easily get this information from the G4ParticleTable object as some of the particles are added to the table on an as-needed basis.
  - One can interrogate the table at the end of a long job but one can never be sure that one has all possible particle definitions.
- Mu2e would like the Geant4 Collaboration to either publish this PDG Particle information in a machine readable way or to provide a description of how to extract complete information in a programmatic way.

Ownership of Pointees

- In many calls to Geant4 methods the user passes an object by pointer.
  - In some cases, Geant4 takes ownership of the object and calls delete on the pointer at the appropriate time.
  - In other cases, the user must call delete.
- There is little documentation: one just needs to know.
- Could there be a method, even structured comments, to indicate which case holds where?
  - It would be OK if this happened over time.