New Developments In Analysis

I. Hrivnacova, IPN Orsay (CNRS/IN2P3)
G. Barrand, LAL (CNRS/IN2P3)

20th Geant4 Collaboration Meeting,
28 September 2015, Fermilab

Outline

- New features
  - Writing/reading histograms at Csv format
  - Batch plotting
  - Handling parameters
  - Handle MPI histograms packing, unpacking
  - Other improvements
- Plans

g4tools

diff –u Okinawa Chicago

G.Barrand, CNRS/IN2P3/LAL

What's new (in one slide)

- Handle MPI histos packging, unpacking.
- Writing/reading histos at csv format released.
- Batch plotting : a nice idea. First code in g4tools. Could be very useful for a "first glance" at the physics output of a batch.
- "usual work" to please Coverity and nightlies ☺

Csv Histograms, Profiles

- Released in 10.1
- Use cases
  - Users that use Mathematica, Matlab or matplotlib (or some other python based tool)
  - Makes possible to switch output format between Root, Xml and Csv without limitations and simplifies testing
- Each histogram/profile is written in its own Csv output file
  - The file name is generated automatically from the base file name set via SetFilename() function or /analysis/setFileName command

Batch graphics

- Be able to produce a .ps containing some graphics (today plots) without having to tie to any graphics external libs (no X11, no GL).
- Done with g4tools/sg classes : it is a scene graph manager. Very flexible and powerful logic to handle graphics.
- It comes with its own zbuffer (then we can do batch 3D !). It is done in pure C/C++ by using CPU, then it would be too slow for interactive but for batch we (quite) don't care.

Batch Graphics

- Histograms and profiles plotting can be activated using G4AnalysisManager functions:

```
// Activate plotting of 1D histogram
analysisManager->SetH1Plotting(id, true);
   // etc for H2, H3, P1, P2
```

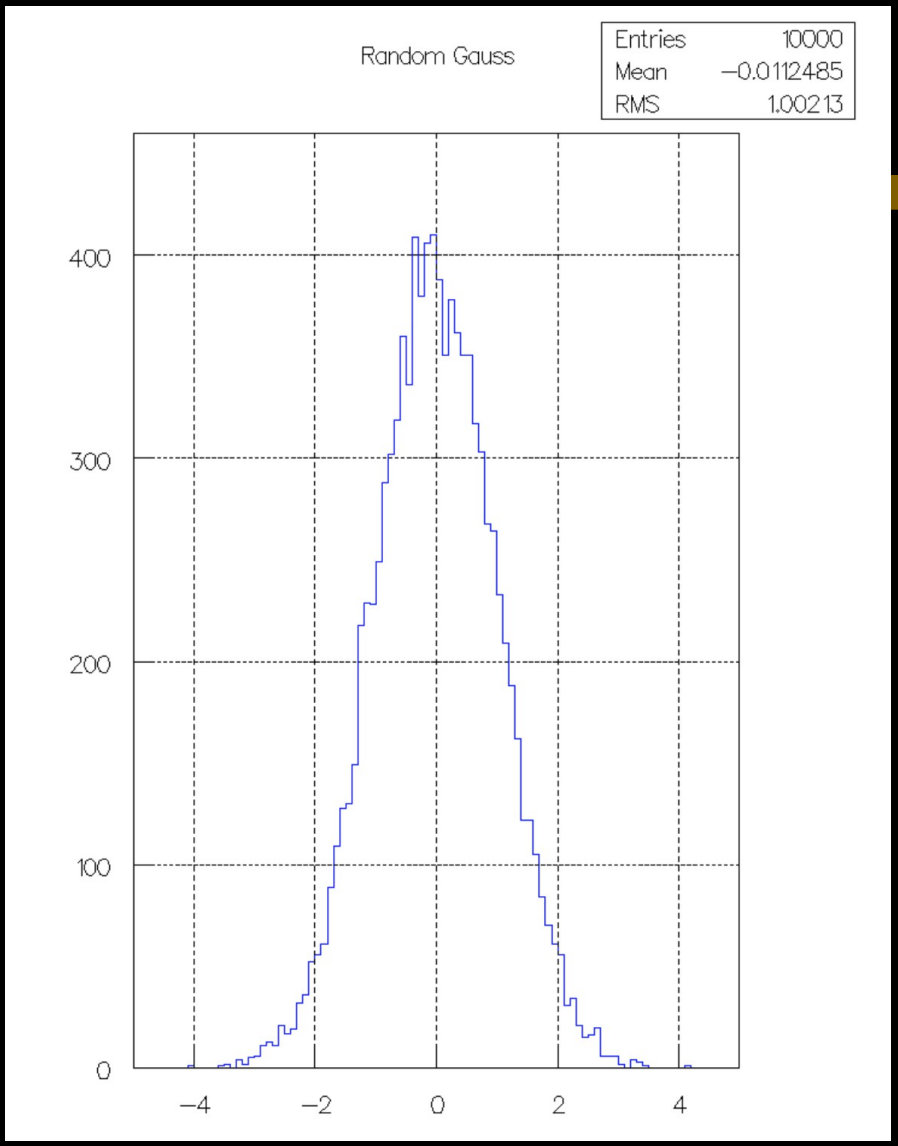
- Or via UI command (still to be implemented)

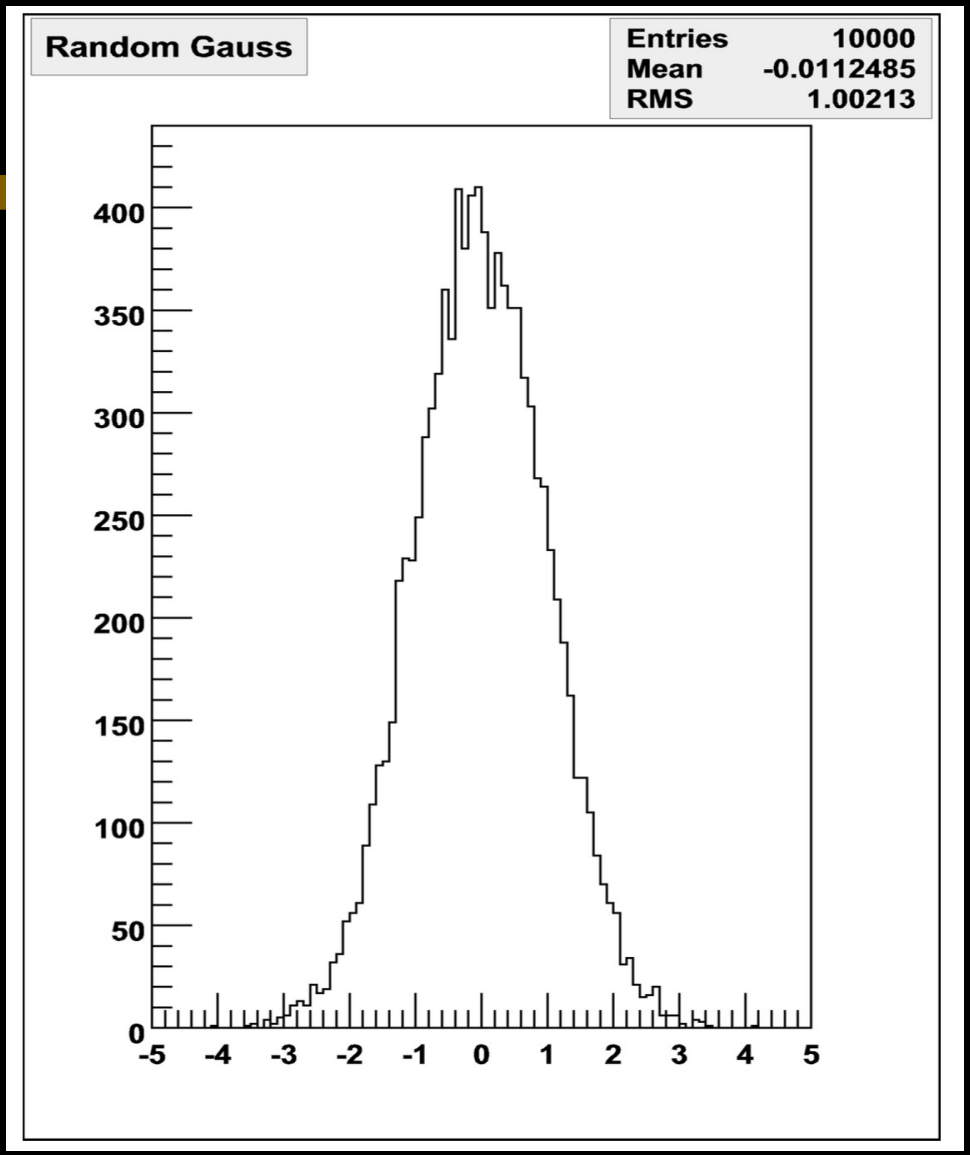
```
/analysis/h1/setPlotting id true|false
/analysis/h1/setPlottingToAll  true|false
   ## etc for h2, h3, p1, p2
```

externals/freetype

- Plotting needs the mastering of text. By default g4tools/sg comes with the “old Hershey HBOOK” fonts done in C.
  (Sufficient for a “first glance” at physics).
- But all is here to render fonts by using the nice freetype2 lib (in fact a C “batch” graphics lib too !) and then reach straight high quality and nice looking plots in pure batch.
- We agreed with the Software Management WG that Geant4 will handle freetype2 as an optional external library via a CMake configure variable, eg. GEANT4_USE_FREETYPE .

HBOOK/freetype





G.Barrand, CNRS/IN2P3/LAL

Handling Parameters

- A wish from Luciano Pandolo (Advanced Examples WG):
- *“Do we have (or do you plan to introduce) the possibility to store numbers (i.e. named parameters)? I am talking about an equivalent of ROOT's TParameter<int>, TParameter<double>, etc.”*
- The motivation is to simplify the basic examples (B1, B3) , used in tutorials, where the novice users have to face introducing the Run class in order to handle a few double data members which have to be merged when running in multi-threading mode
- First implementation is now being discussed with Luciano and Michel Maire (responsible of B1 and B3 examples)

B1 example

```
class B1Run : public G4Run {
  public:
    ...
    // method from the base class
    virtual void Merge(const G4Run*);
    void AddEdep (G4double edep);
    // ...
  private:
    G4double  fEdep;
    G4double  fEdep2;
};
```

```
#include "G4Parameter.hh"
...
class B1RunAction : public G4UserRunAction {
  public:
    ...
    // method from the base class
    void AddEdep (G4double edep);
    // ...
  private:
    G4Parameter<G4double>  fEdep;
    G4Parameter<G4double>  fEdep2;
};
```

Run class and Merge() method are not needed

B1 example (cont.)

```
#include “G4ParameterManager.hh”
...
B1RunAction::B1RunAction()
: G4UserRunAction(),
  fEdep("Edep", 0.),
  fEdep2("Edep2", 0.)
{
  //Register parameter to the parameter manager
  G4ParameterManager* parManager = G4ParameterManager::Instance();
  parManager->RegisterParameter(fEdep);
  parManager->RegisterParameter(fEdep2)
}


void B1RunAction::EndOfRunAction(const G4Run* run) {
  ...
  // Merge parameters
  G4ParameterManager* parManager = G4ParameterManager::Instance();
  parameterManager->Merge();
  ...
}
```

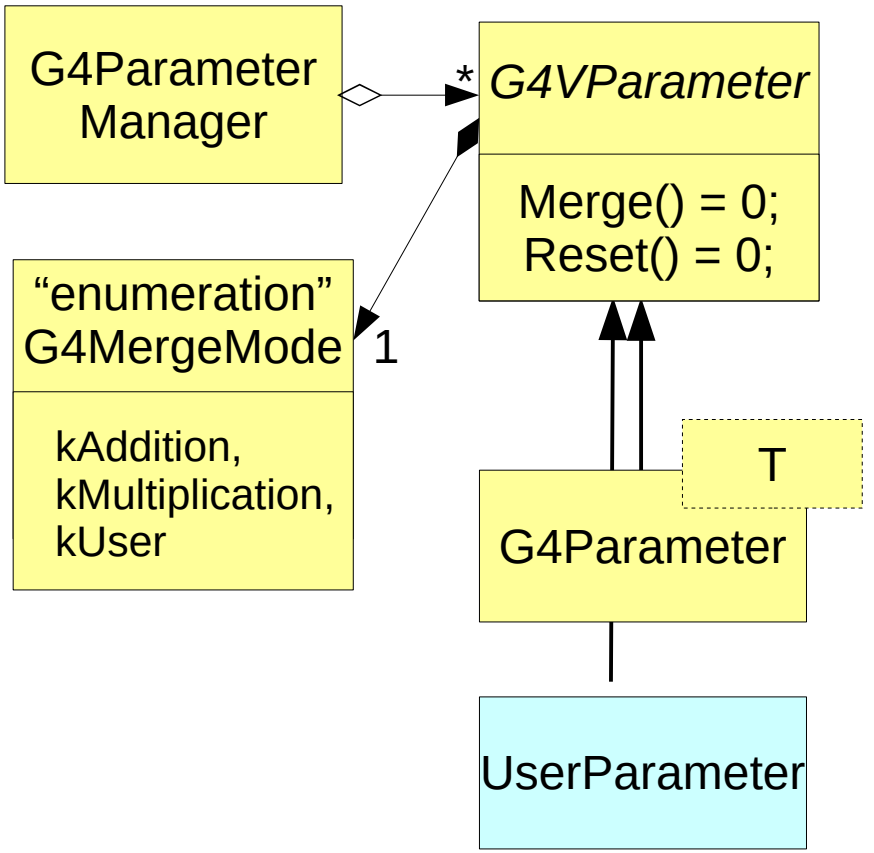
The parameters are initialized with a name and a value

The parameters not created via the manager have to be registered to it

The call to Merge() may be not necessary if we hook the parameter manager to G4AnalysisManager

Handling Parameters
Classes

- G4ParameterManager is a (thread local) singleton
  - Independent from other analysis managers
  - Has std::map<G4String, G4VParameter*>
- Provide functions both to create and to register a parameter
- Performs Merge() of all parameters
  - The merge mode can be selected per parameter
- Users can define their own parameters
  - Tested with std::map<G4String, G4int> used for processes counting in TestEm* examples



MPI

Development in collaboration with Andrea Dotti

- In g4tools:
  - tools::histo::hmpi: interface to MPI not requiring MPI installation
  - tools::mpi::hmpi: the MPI implementation of the histo::hmpi interface
- In analysis:
  - New G4AnalysisManager function for MPI merging:

    G4bool Merge(tools::histo::hmpi* hmpi);
  - This keeps analysis category independent from MPI libraries
- In examples/parallel/MPI:
  - New G4MPIhistoMerger class, which replaced previous implementation based on own implementation of merging the tools objects
  - Merging is now performed using G4AnalysisManager with tools::mpi::hmpi
  - No need to access the tools/g4analysis internals

Other Improvements

- Most of C++11 features recommended by C++11 task force were applied in the analysis category classes
  - Auto, range-based loops, nullptr, alias declarations, scoped enums, deleted/overriding functions, explicit constructors
- Templates were introduced where suitable to avoid code duplications
  - They concern only analysis classes internals and do not affect the API seen by the users
- The test code in test03 is being constantly updated with new features
  - Added Hbook test
- Extension of UI commands for setting histograms and profiles parameters per axis
- Added activation/inactivation mechanism for ntuples

Plans

- Batch plotting
  - Allow user customization of the plot configuration
  - Looking for an automated way how to propagate features from tools to analysis and Geant4 UI
- Requested by users
  - Sparse histograms – in Work plan
  - Handling more files by analysis manager – still to be considered
  - Provide an example of usage of ntuple columns of vector type - promised
- Stop support for HBOOK output after 10.2
  - The development and support for CERNLIB is stopped at CERN
  - The binaries are not provided for new platforms
- Continue addressing new requests from users