GEANT4 navigation with ROOT/TGeo

Andrei Gheata (CERN/ALICE)

AA meeting 13 December ‘06

Outline

- Motivation
- TGeo@VMC
- Implementation
- How it works
- Ongoing tests & validation procedure
- Integration in GEANT4 VMC
- Conclusions

Motivation

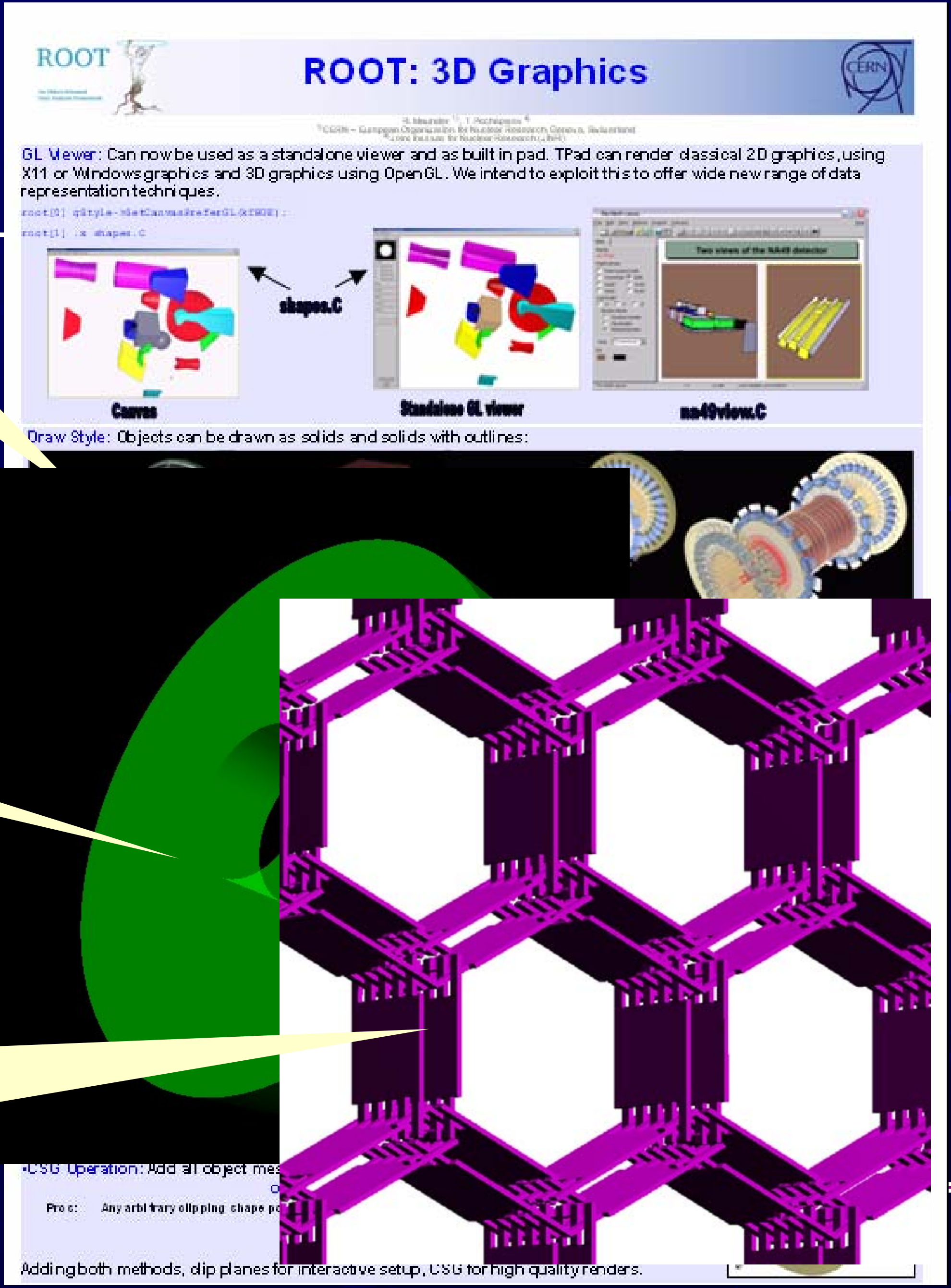
- Running GEANT4 MC with ROOT geometry modeller
  - ROOT TGeo is a multi-purpose geometry engine independent from any MC framework
    - Developed by ALICE offline in collaboration with ROOT team
    - Optimized/tuned for navigation performance in detector geometries
    - Additional features: checking tools, visualization, alignment tools, ROOT persistency, …
- Having an external geometry description as alternative to MC-embedded models
  - Geometry is used a lot outside the simulation framework
  - GEANT3 and FLUKA were already interfaced with TGeo
  - Important component in making a simulation application survive outside the specific MC framework → Virtual Monte Carlo

Can take advantage of graphics accelerators

Automatic illegal overlap detection

Run-time volume assemblies

Misalignment of ideal geometry



ROOT: 3D Graphics

GL Viewer: Can now be used as a standalone viewer and as built in pad. TPad can render classical 2D graphics, using X11 or Windows graphics and 3D graphics using OpenGL. We intend to exploit this to offer wide new range of data representation techniques.

```
root[0] gStyle->SetCanvasPreferGL(kTRUE);
root[1] .x shapes.C
```

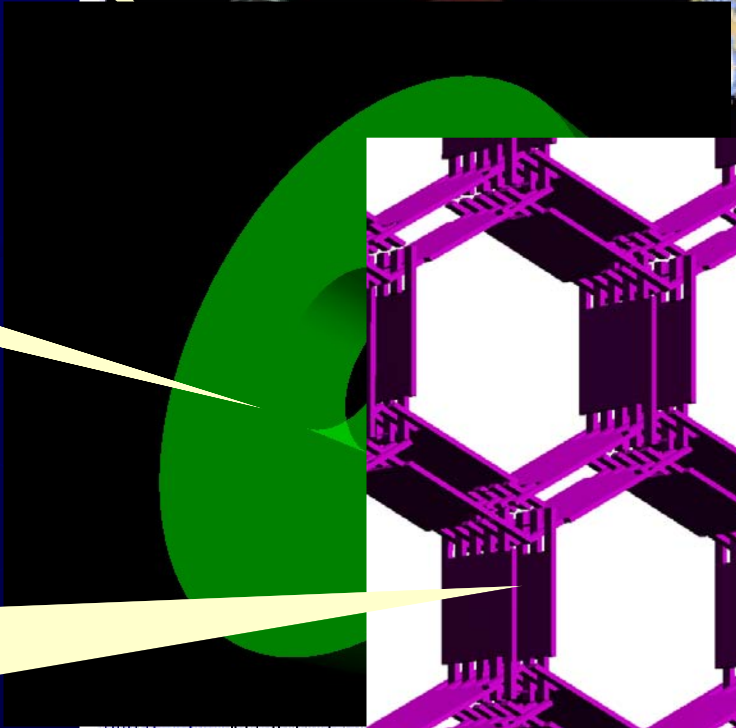
shapes.C
Canvas
Standalone GL viewer
na49view.C

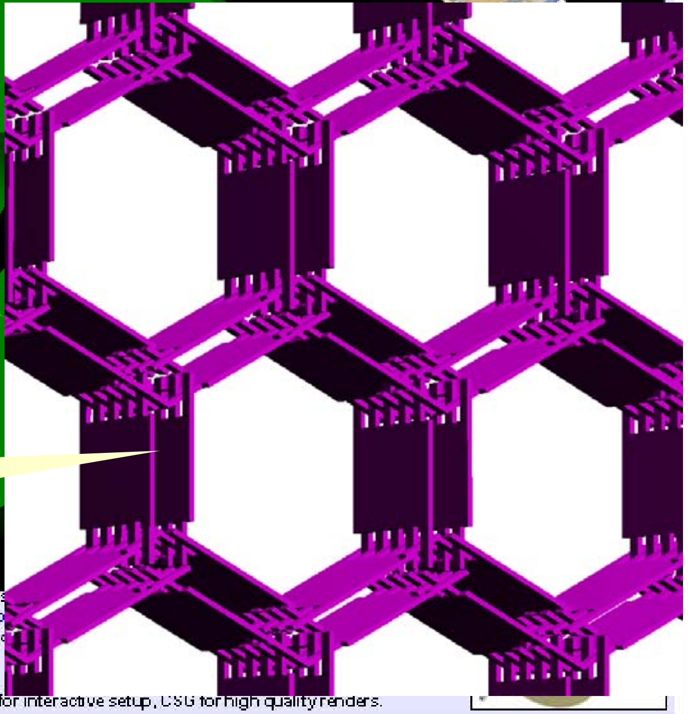
Draw Style: Objects can be drawn as solids and solids with outlines:

•CSG Operation: Add all object mes

Proc: Any arbitrary clipping shape po

Adding both methods, clip planes for interactive setup, CSG for high quality renders.

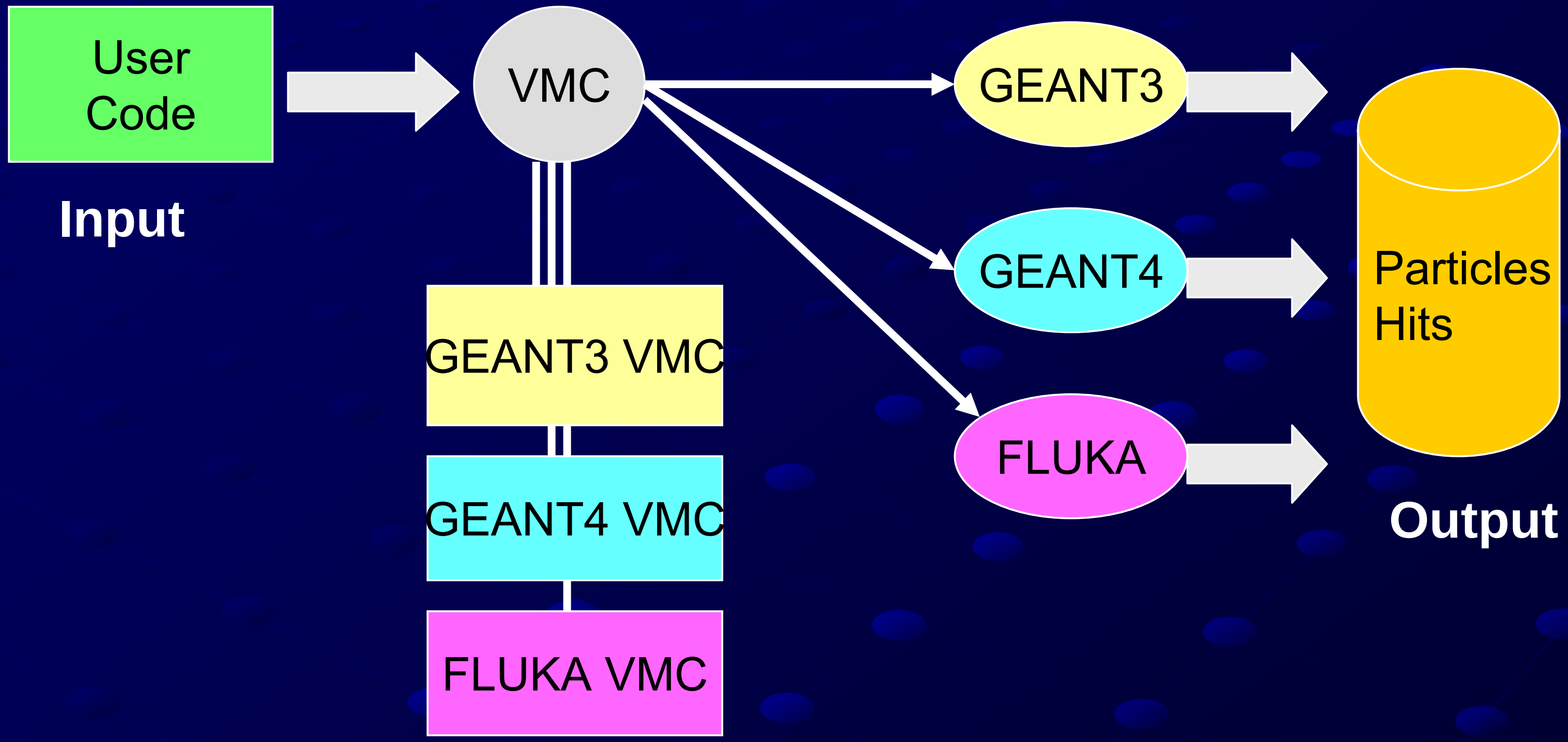




The Virtual Monte Carlo idea

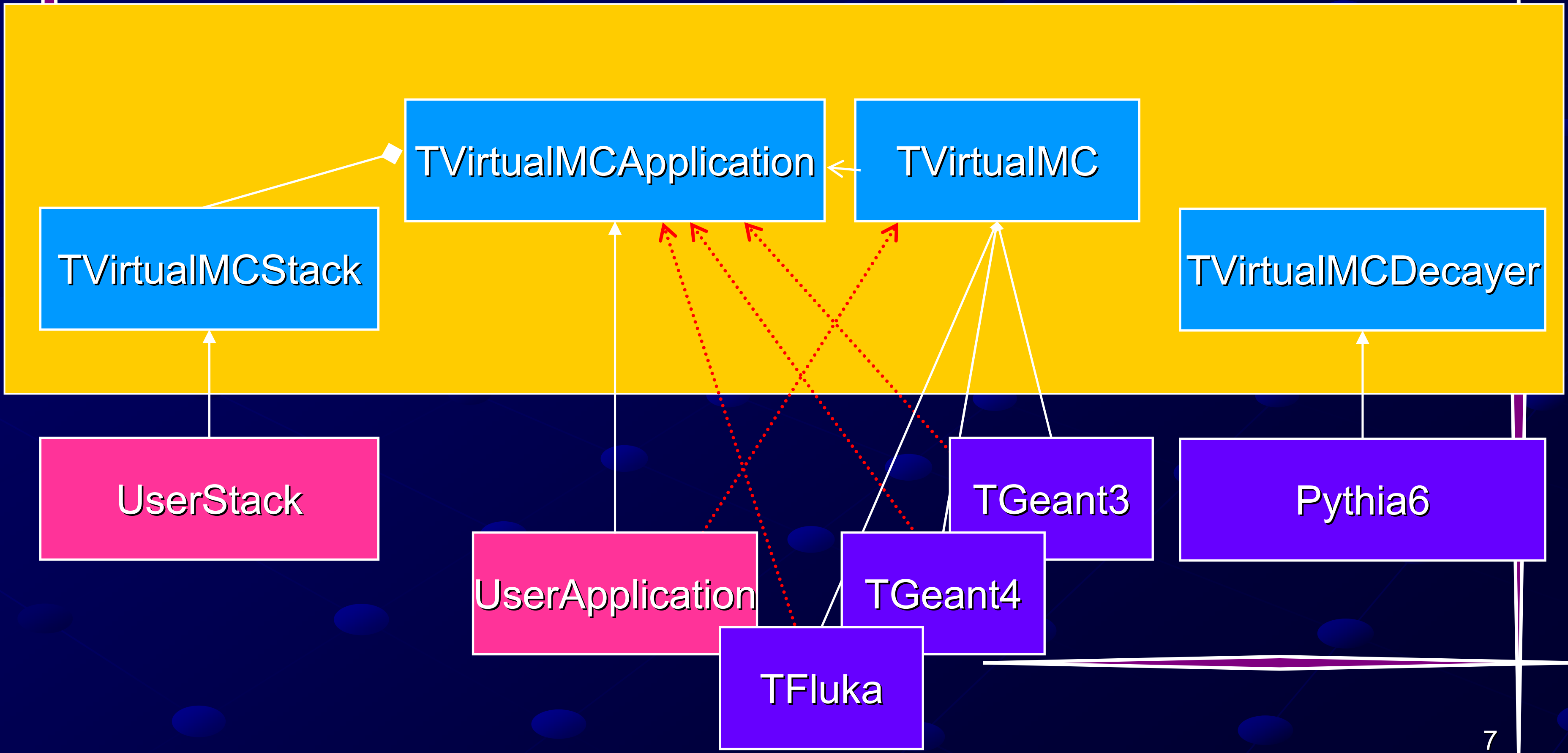
- Make user application independent of specific transport code
  - GEANT3, GEANT4 and FLUKA supported so far
  - Keeping the SAME definition for geometry, I/O formats and detector response simulation
    - Allows running the same application with different MC's
- VMC decouples user code from the concrete MC
  - Provides MC-independent API for :
    - Geometry definition, setting up physics, setting up cuts, handling particle stack … (features very different from one MC to another)
    - Querying MC machine state and kinematics during stepping callbacks (very similar mechanisms)

VMC concept


User Code
Input
VMC
GEANT3 VMC
GEANT4 VMC
FLUKA VMC
GEANT3
GEANT4
FLUKA
Particles
Hits
Output

VMC design

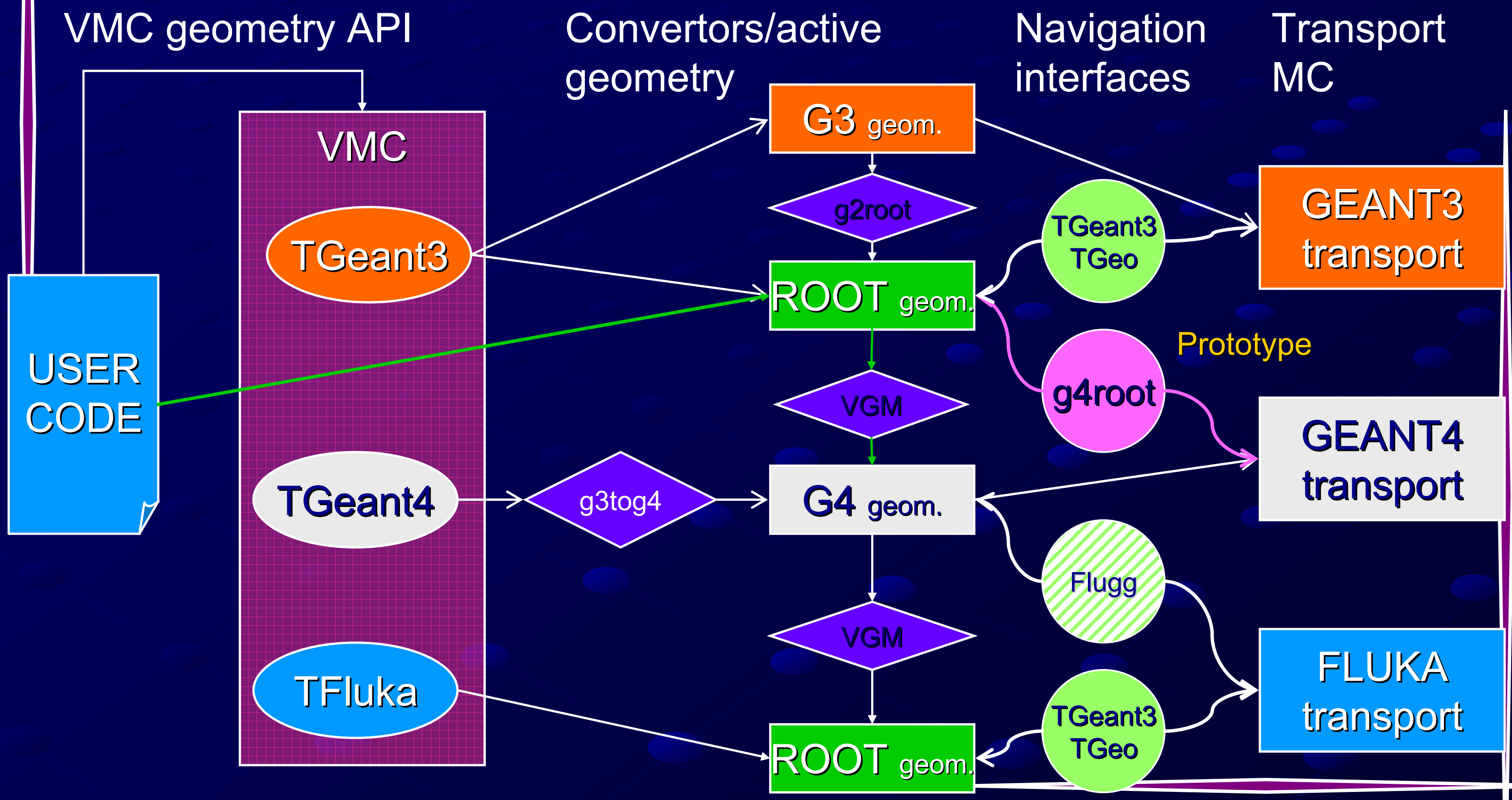
- Communication only via interfaces



Geometry in VMC

- Geometry models are quite different for different MC's
  - There is a common denominator G3/G4
  - FLUKA geometry completely different
  - What do we do to provide the geometry that transport MC wants?
- There are several geometry convertors, working to a certain extent
  - g2root (G3 → TGeo), g3tog4 (G3 → G4)
  - Most advanced: VGM (G4 ↔ XML (AGDD, GDML) ↔ TGeo)
    - http://ivana.home.cern.ch/ivana/VGM.html
- VMC provides API for geometry creation
  - Inspired by GEANT3 – supports only G3 features

VMC geometry – supported options

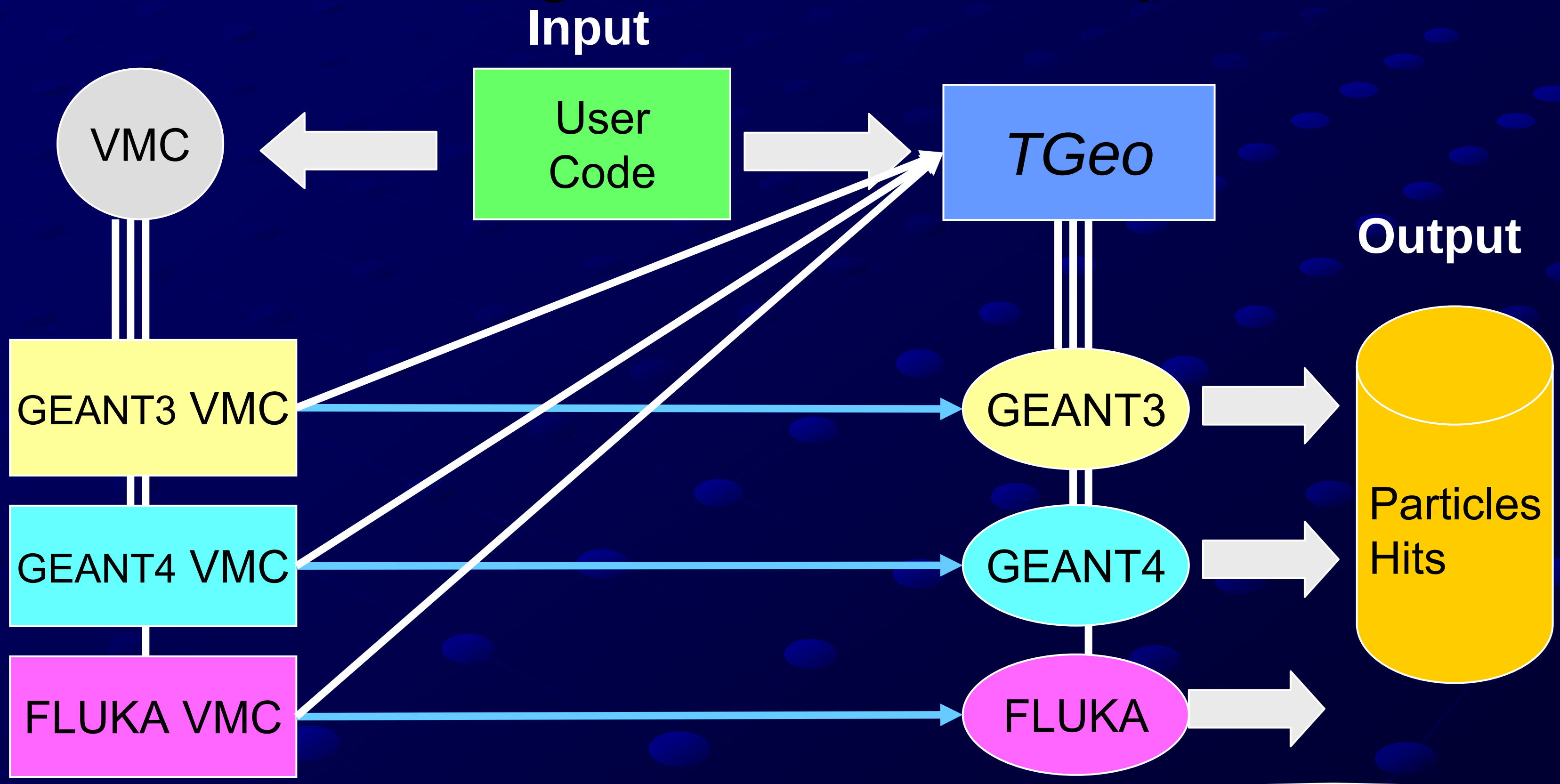


External geometry – a coherent solution

- Experience show that all conversions have serious limitations
  - Not all features can be converted
    - Matters a bit when results get different due to geometry differences
    - Matters a lot when not all geometry can be converted
- One of the reasons TGeo was developed
  - Navigation speed was a priority
    - Even behind an interface TGeo gains ~8% in simulation speed in case of ALICE compared to GEANT3 native
      - Even more in case of non-optimized geometries
- In case of FLUKA there were no other way out for running ALICE simulation
- **We needed to make all supported MC transport engines able to work directly with TGeo**

TGeo@VMC concept

- TGeo as navigator for all transport MC's



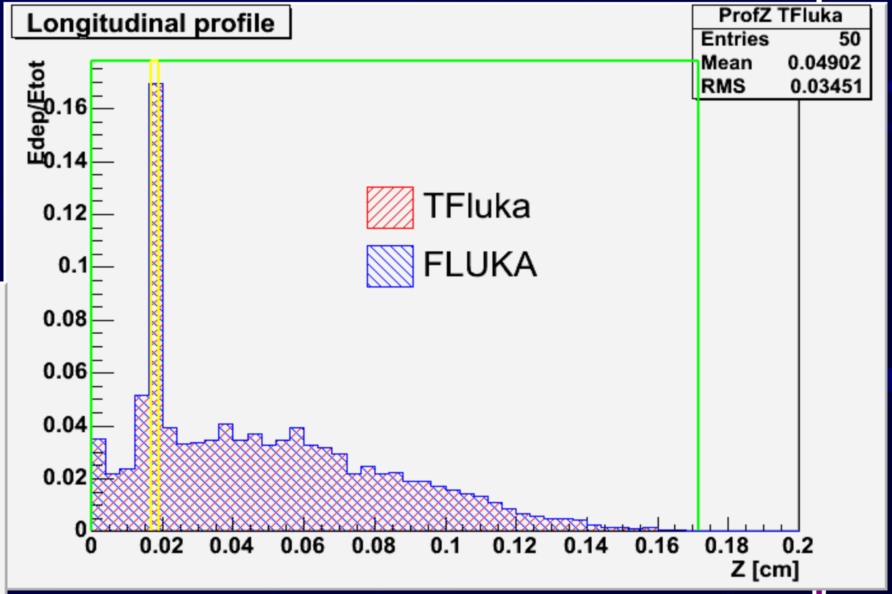
Navigation interfaces

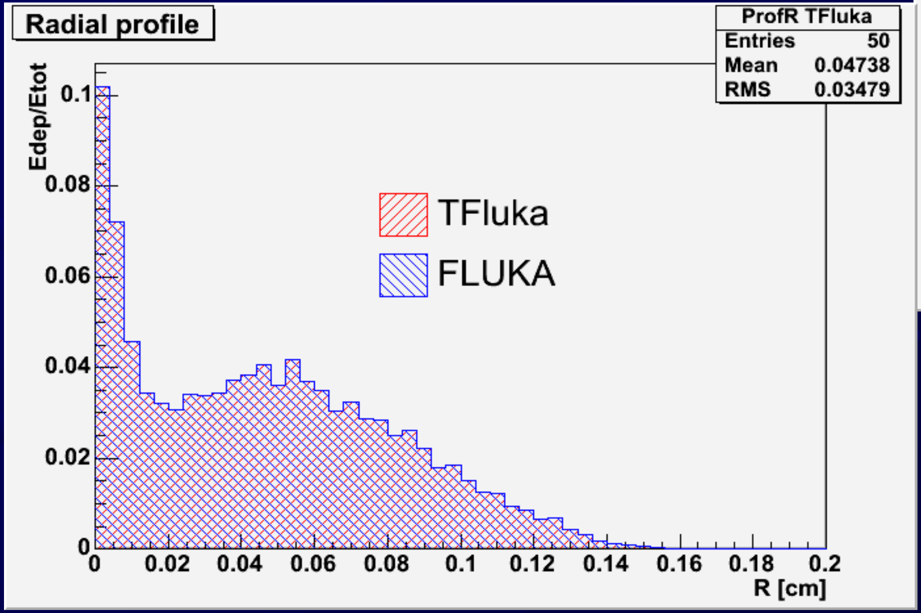
- Interface mechanisms providing geometry answers to queries posted by transportation
  - Thin layer based on wrapper functions/ virtual methods “spying” the geometry channels
- GEANT3 – Geant3TGeo interface
  - Validated
- FLUKA – Fluka MC Geometry interface
  - Validated
- GEANT4 – G4ROOT interface
  - 2 modes supported:
    - Via VMC
    - With a native G4 application with geometry defined via Root (or converted to ROOT structures)
  - Ongoing tests

How the navigation validation was done…

FLUKA electron transport in thin layers

- AlAuAl FLUKA native example
- 1000 electrons at 1 MeV, EM cascades
- Same final random number after simulations with FLUKA native and TFluka
- The same for all 3 tested examples



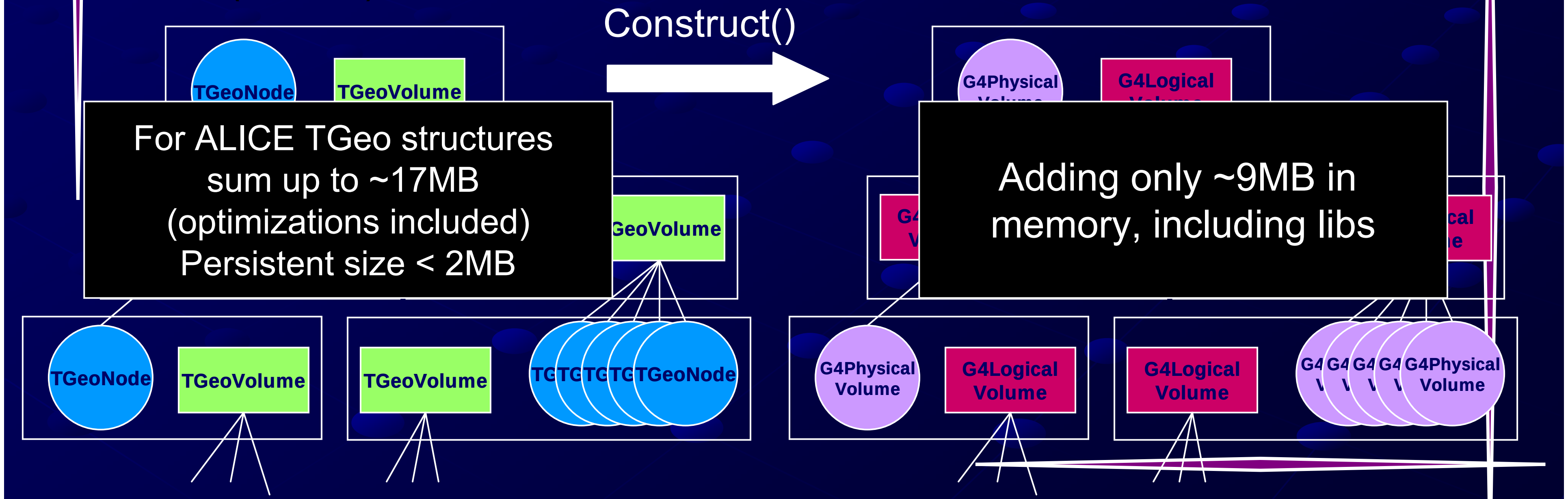


Implementation of G4 navigation interface

- GEANT4 transportation as well as several physics processes need real G4 geometry objects
  - Sensitive detectors, user limits, optical properties are connected to G4LogicalVolume
  - Need to provide geometry states that are understandable by G4
- TG4RootDetectorConstruction : public G4VUserDetectorConstruction
  - Having a TGeo geometry in memory (TGeoManager) is a prerequisite
  - *Construct()* method building up a G4 hierarchy corresponding to TGeo one

Implementation – building the structure

- Special user detector construction registered to G4RunManager
  - *Construct()* method called during initialization
  - G4 logical hierarchy created, but no optimization structures (voxels) needed



For ALICE TGeo structures sum up to ~17MB (optimizations included) Persistent size < 2MB

Adding only ~9MB in memory, including libs

Implementation – navigation interface

- G4Navigator – base class for GEANT4 navigation
  - Very useful help from GEANT4 team in changing some methods/data members qualifications – made the implementation easier
- TG4RootNavigator : public G4Navigator
  - Implementing methods for locating a point, finding the distance to next boundary, computing the safety distance and the normal to a crossed surface

Implementation – connecting to G4 geometry

- Several user-defined objects need to be connected to the newly-created G4 volumes
  - Sensitive detectors manager, user limits, …
  - G4 geometry objects are mapped to TGeo ones and accessible form TG4RootDetectorConstruction
- TVirtualUserPostDetConstruction
  - Pure virtual plugin class
  - *Initialize()* user method to implement for connecting additional objects to geometry

Interface management

- Root-ified manager class allowing connection to G4
  - TG4RootNavMgr : public TObject
- Additional class interfacing TGeo shapes as G4 solids:
  - TG4RootSolid : public G4VSolid
  - Navigation methods of solids not used in the interface
    - Navigation entirely redirected to TGeo
  - Possibility to use directly TGeoShape objects in a native G4 geometry
    - Few methods not yet implemented at this level
    - The feature will be supported by the interface

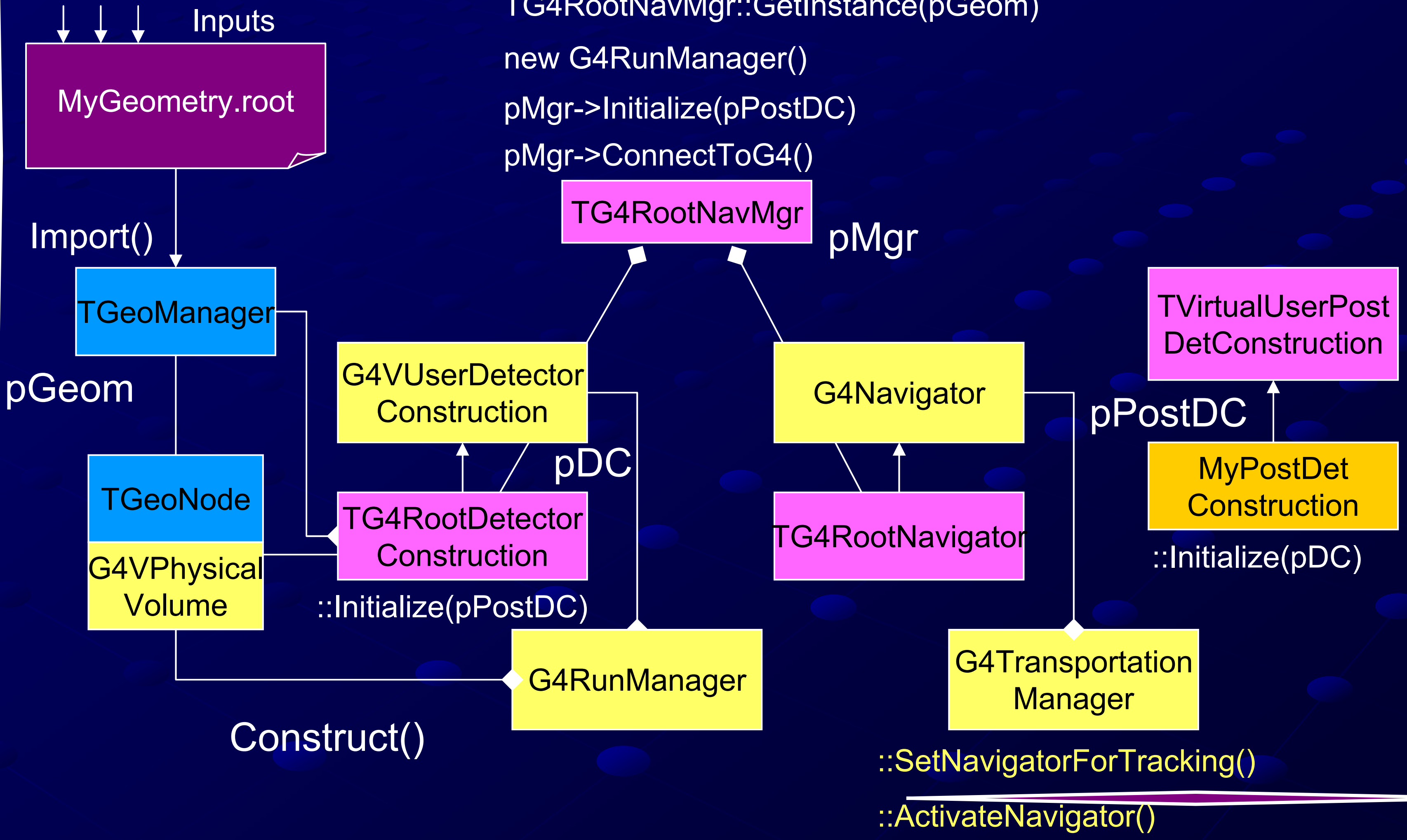
How it works

TG4RootNavMgr::GetInstance(pGeom)

new G4RunManager()

pMgr->Initialize(pPostDC)

pMgr->ConnectToG4()



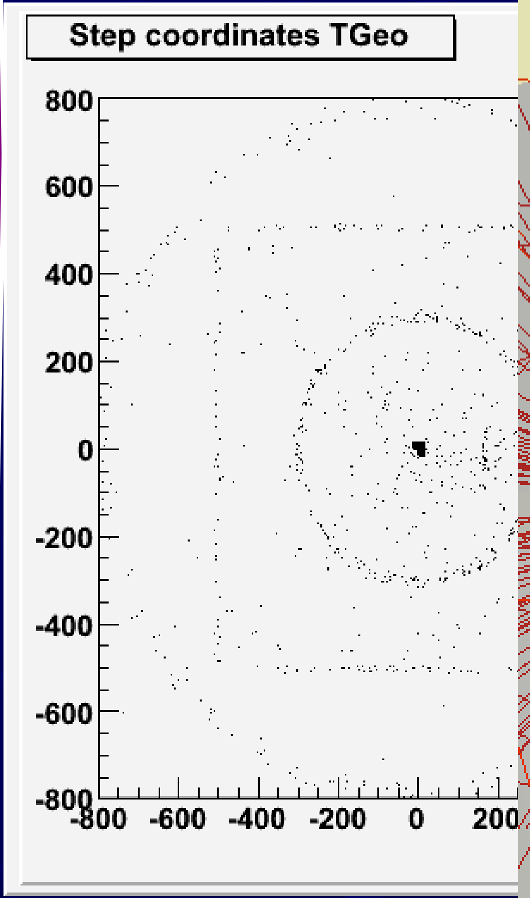
Availability

- Interface introduced in CVS HEAD of ROOT as G4ROOT module
  - README file describing how to compile the module
  - Can only be compiled with HEAD development version of GEANT4
    - For public use, available from 15 December:
      - ROOT version 5.14
      - GEANT4 version 8.2
- Direct comparison tests between normal/g4root navigation performed on modified versions of GEANT4 novice examples
  - One (N06) included in G4root/test

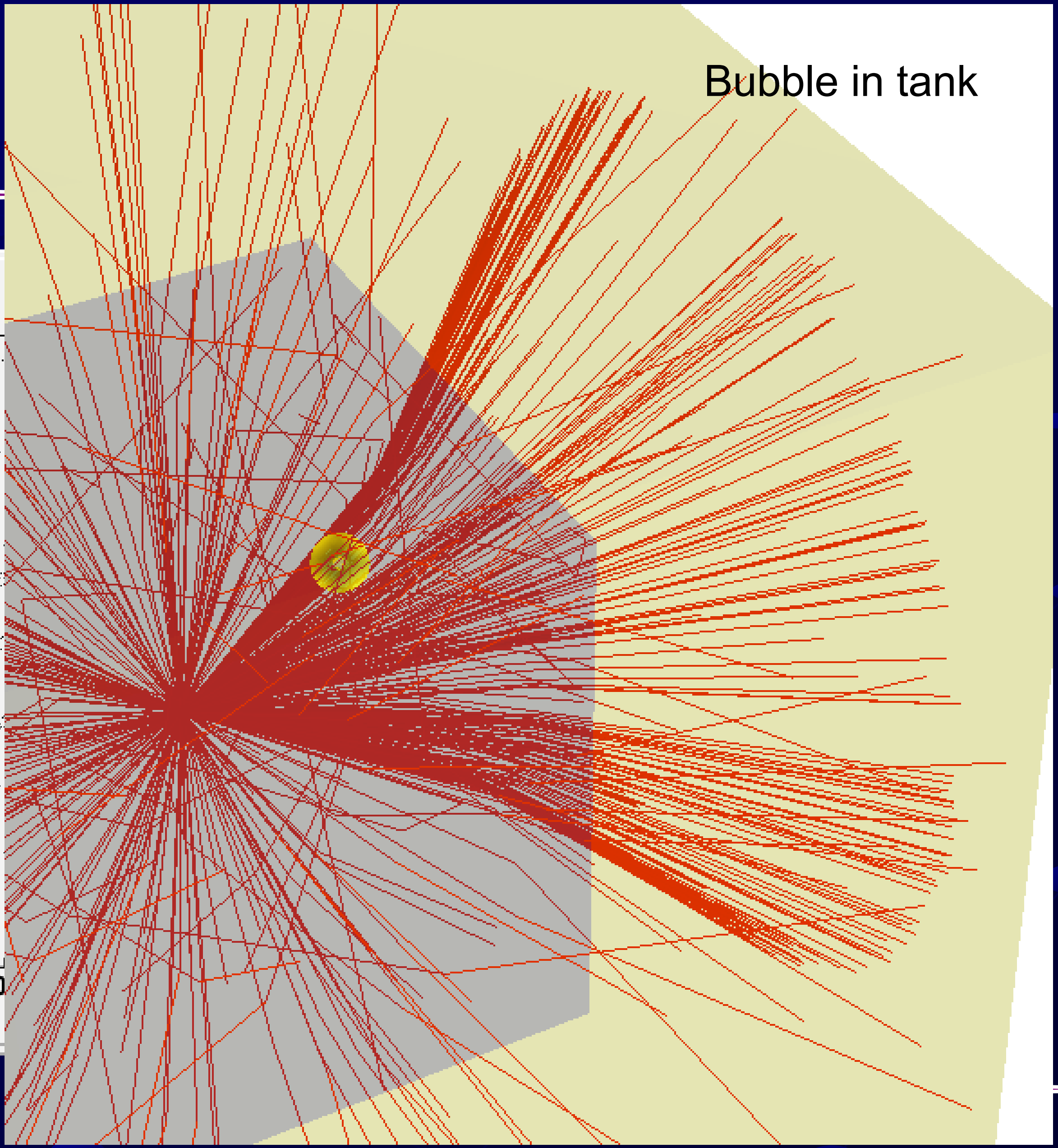
Example: optical photon transport

- Modified version of Novice example N06
- Adds few files to N06 ones
  - Demonstrating the usage of the interface
- Comparing results when running with G4 or G4ROOT navigation
  - Geometries used in the 2 cases equivalent
  - Same initial random seed
  - Comparison plots of XY coordinates at boundary crossings
- Showing a graphical output of tracks over the geometry

Test outp



Step coordinates TGeo
800
600
400
200
0
-200
-400
-600
-800
-800
-600
-400
-200
0
200



Bubble in tank

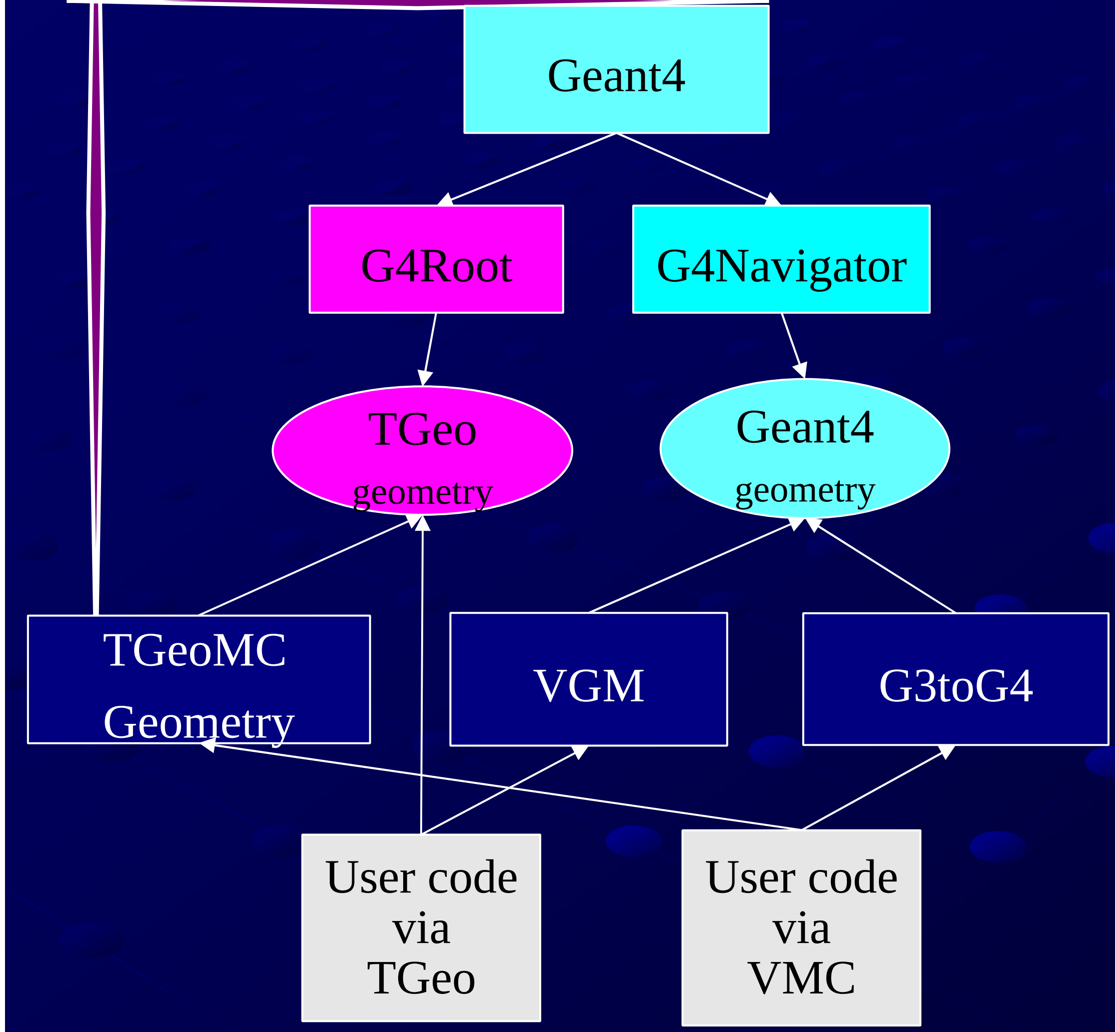
Support for TGeo navigation in GEANT4 VMC

- Support for TGeo geometry
  - Since v1.0 (Jul 2003) - via roottog4 converter
  - From v1.6 (Mar 2005) – via VGM (Virtual Geometry Model, the generalized version of roottog4 converter)
  - Version 2.0 (in CVS) – direct use of TGeo geometry in Geant4 via G4Root
    - Tested on the VGM geometry tests & VMC examples
    - Identical results for tracking with geantinos
    - Differences in tracking of physical particles – under investigation
    - To be released just after Geant4 and Root releases in Dec 2006
  - To be tagged after ROOT/G4 releases
  - Several ongoing tests for consistency
- By Ivana Hrivnacova, IPN Orsay

GEANT4-VMC run configuration

- User has a choice to define geometry via:
  - VMC – GEANT3 style
  - TGeo geometry modeller in ROOT
  - Geant4 - standard Geant4 detector construction class
- User can choose to use G4 native or G4ROOT navigation.
- Selection in the run configuration object instantiated in the configuration macro
  - VMCtoGeant4
  - VMCtoRoot
  - RootToGeant4
  - Root
  - Geant4
- The first word means geometry input, the second one navigator to be used

GEANT4 VMC run configuration



Geant4 VMC connects automatically necessary packages and activates selected navigator

With both geometry definition via VMC and TGeo user has a possibility to run with both G4 native and G4Root navigation

Using G4ROOT from a native G4 application

- Can this be done ?
  - Yes, an example of use G4Root with G4 native application with geometry defined via Root is provided with G4Root package
  - The sensitive detectors and other user are connected to geometry in a bit different way as it is demonstrated on the example objects to the generated G4LogicalVolume objects
  - Possibility to convert G4 geometry to TGeo
    - Most G4 geometry features existing also in TGeo
    - Procedure not yet automated
    - Could be avoided in future by keeping the original G4 geometry
- Does it worth trying ?
  - Yes, a minimal working setup for checking performance issues can be put together quite easily
    - Can give fast an idea if the interface usage can save some CPU

Validation procedure

- Testing with G4 native examples (M.Gheata)
  - Comparing results given by G4/G4ROOT navigation
  - Equivalent TGeo geometries with G4 ones
  - Done for some Novice examples
    - Most give identical results (N01, N03, N05, N06)
    - Some differences (random seed change) observed for N02
      - Under investigation – possibly due to a geometry difference or difference in safety distance evaluation
- Tests within VGM test setups (I. Hrivnacova)
  - Checking various geometry configurations and different geometry features
    - Boolean operations, divisions, replicas, …
  - Revealed few bugs in the navigation interface that were fixed

Validation procedure (cont)

- Tests with GEANT4 VMC
  - Novice examples with either G4 or G4ROOT navigation available in G4_VMC (thanks to I.Hrivnacova)
    - Some differences in results need to be further understood
  - Comparisons with physics switched on/off to be done for real experiments (to be done)
    - ALICE will be the first candidate
    - Further testing/feedback expected from MINOS (S. Kasahara), CBM/PANDA@GSI/FAIR
  - Performance comparisons for complex geometries (to be done)
    - TGeo navigation performance demonstrated against GEANT3 (faster by 8%
    - Realistic timing comparisons can only be done for realistic cases (example geometries much too simple)
    - Performance-wise optimizations not yet done for the interface

Conclusions

- A prototype of the navigation interface of GEANT4 using TGeo geometry available
  - Starting with ROOT v5.14, GEANT4 v8.2
  - Interface used by GEANT4_VMC v2.0
- The interface can be used ‘as is’ by G4-based applications with a TGeo geometry representation and minor additions (example available)
- Most benefits by usage via VMC
  - User application kept unchanged for 3 MC’s
  - Navigation based on the same geometry allow reliable comparisons for the MC predictions

Acknowledgements

- GEANT4 team
  - Support for changes in G4Navigator to accommodate this interface
- ALICE Offline & ROOT teams
  - Supporting the development and facilitating the integration in ROOT framework
- All current (and future) testers…