ALICE Simulation Validation

I.Hřivnáčová, IPN Orsay
For the ALICE Collaboration

LPCC Detector Simulation Workshop,
18 -19 March 2014, CERN

Outline

- ALICE Offline Framework
  - O^2 for Run 3
  - Virtual Monte Carlo
- Geant4 Validation
  - Physics validation
  - Performance Optimization
- New ways to speed up
  - Multi-threading
  - Possible improvement of current simulation setup
- Fast Simulation

The ALICE offline framework

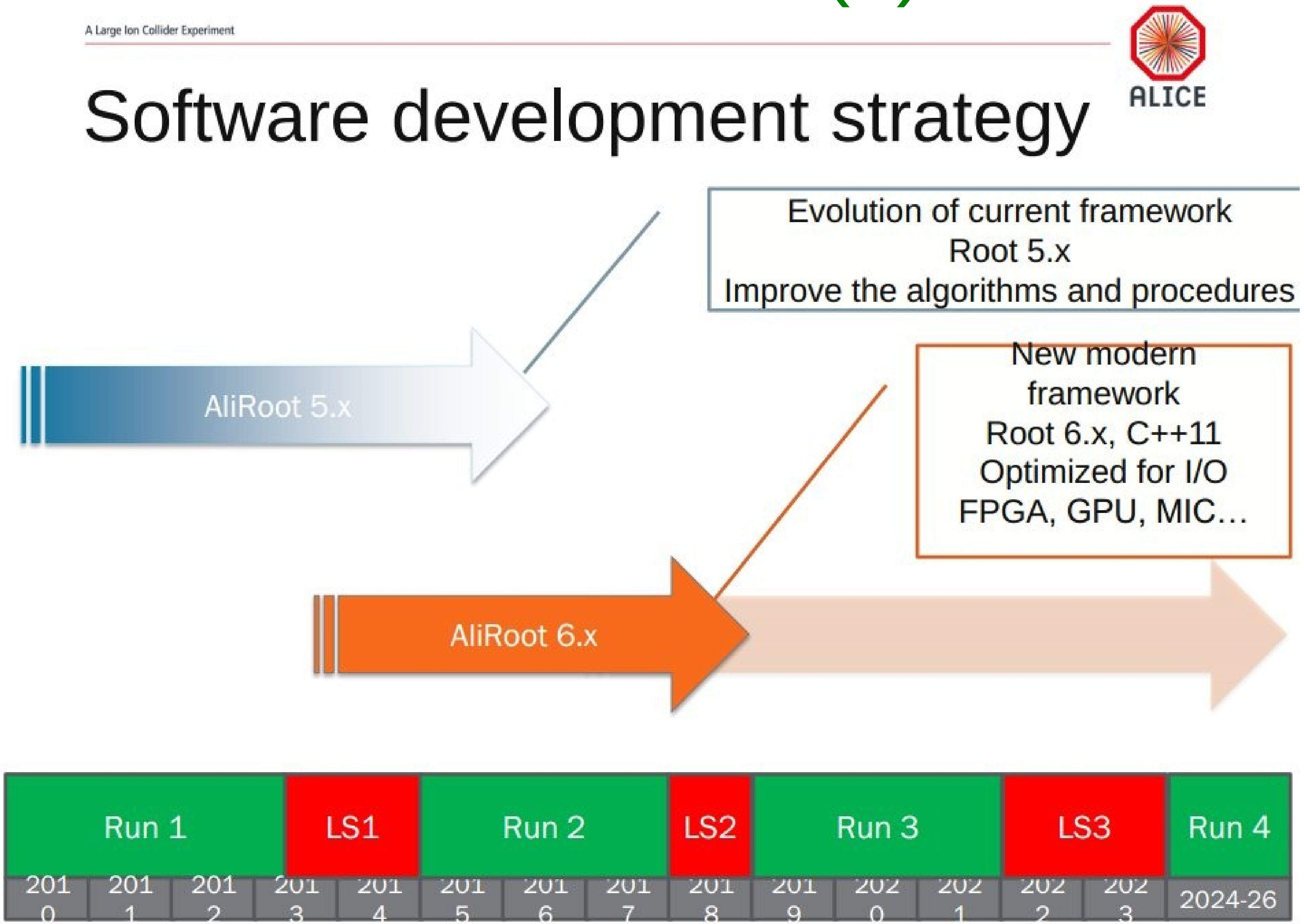
- AliRoot = the ALICE offline framework
  - The framework for simulation, reconstruction and analysis
  - Uses the ROOT system as a foundation
  - Based on the Object Oriented programming paradigm, and is written in C++ - except for large existing libraries, such as Pythia6, HIJING, and some remaining legacy code
- Developed since ~ 1998
  - Actual version: 5-04-Rev18
  - Current simulation base on Virtual Monte Carlo framework used in production with Geant3
- AliRoot 6.0
  - New development line based on new core software

O^2 for Run3

- Common software project for ALICE Online + Offline + HLT for software development for Run 3
  - Challenges: both event rate and data volume increase
  - Solution: Moving part of Offline processing to Online
- Overall schedule:
  - Started in Sep 2012 with ALICE Upgrade LoI
  - Mar 2013: Established O^2 Computing Working Groups, now ~13 CWG
  - Sep 2014: O^2 Technical Design Report
- CWG8 = Physics simulation

  Participating institutes [number of people involved]:
  - Indian Institute of Technology, Mumbay [5], CERN [5], Oak Ridge National Lab (ORNL) [2], University of Tennessee [1], Wayne State University [1], INFN [2], University Heidelberg [1], IPNO (IN2P3/CNRS, Univ-Paris-Sud) [1], University Lund [1]

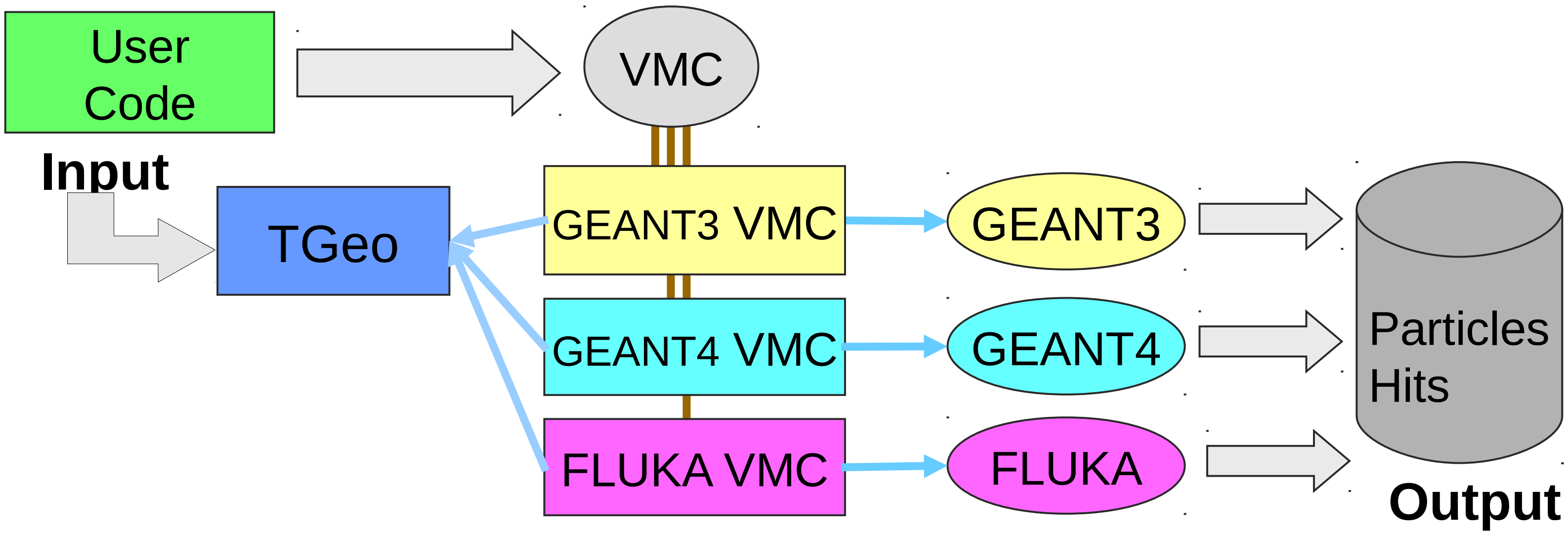
O^2 for Run3 (2)



A Large Ion Collider Experiment
ALICE
Software development strategy
Evolution of current framework
Root 5.x
Improve the algorithms and procedures
New modern framework
Root 6.x, C++11
Optimized for I/O
FPGA, GPU, MIC…
AliRoot 5.x
AliRoot 6.x
Run 1
LS1
Run 2
LS2
Run 3
LS3
Run 4
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024-26

Virtual Monte Carlo

- AliRoot 5 Simulation is based on the Virtual Monte Carlo interface (VMC)
  - VMC is distributed with ROOT and is used in more experimental frameworks
  - http://root.cern.ch/drupal/content/vmc
- VMC allows to run simulation with different Monte Carlo codes from the same user application



Geant3 / Geant4

- Current ALICE simulation based on Virtual Monte Carlo framework
  - Can be run with both Geant3 and Geant4
- Geant3 is used in production
- Geant4 still only in test production
  - Physics results have been validated in 2010 – 2011 for most detectors, TPC simulation in 2013
  - The problem with speed: Geant4 simulation ~ 3 times slower than Geant3
- FLUKA has been also interfaced but the authorization of its usage with VMC was restricted to ALICE

Geometry

- VMC is integrated with TGeo geometry modeller (ROOT)
  - http://root.cern.ch/download/doc/ROOTUsersGuideChapters/Geometry.pdf
- Geant4 VMC supports
  - Root geometry (TGeo), VMC geometry, Geant4 geometry as input
  - Geant4 native navigation - if geometry defined via Root, a conversion from Root to Geant4 is performed via VGM (Virtual Geometry Model)
    - http://ivana.home.cern.ch/ivana/VGM.html
  - G4Root navigation – for geometry defined via Root or VMC
    - Developed and maintained by A. Gheata (CERN)
- ALICE is using TGeo geometry + G4Root navigation

Geant4 Validation

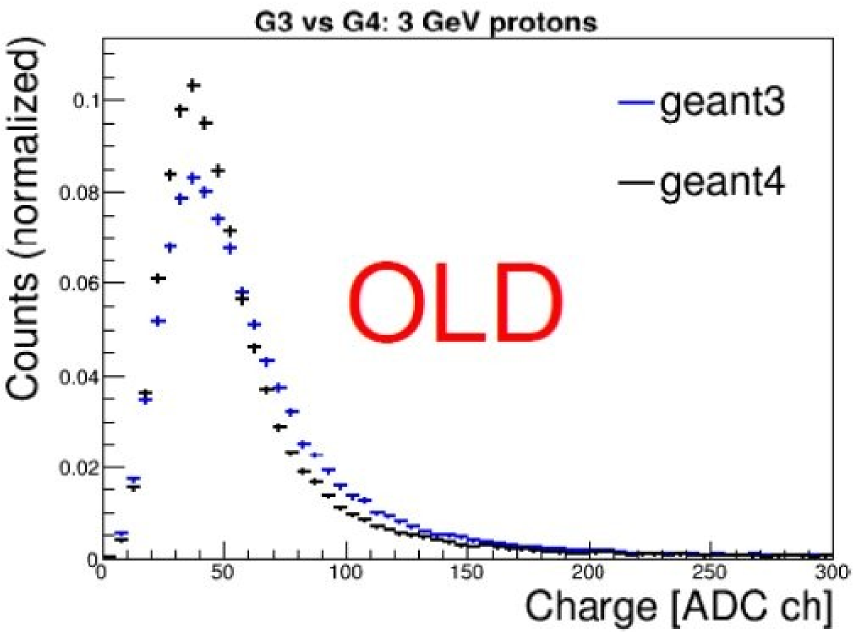
- Test productions ran on the GRID in 2010-2011
  - pp @ 7 TeV, Nuclei cocktail production
  - With Geant4 9.3.x and Geant4 9.4.x and Geant4 9.5.b01
  - Physics lists: QGSP_BERT, QGSP_BERT_EMV, QGSP_BERT_CHIPS, QGSP_FTFP_BERT
- No central Geant4 test productions were run since this time, but Geant4 and Geant4 VMC are installed on GRID and available for individual use

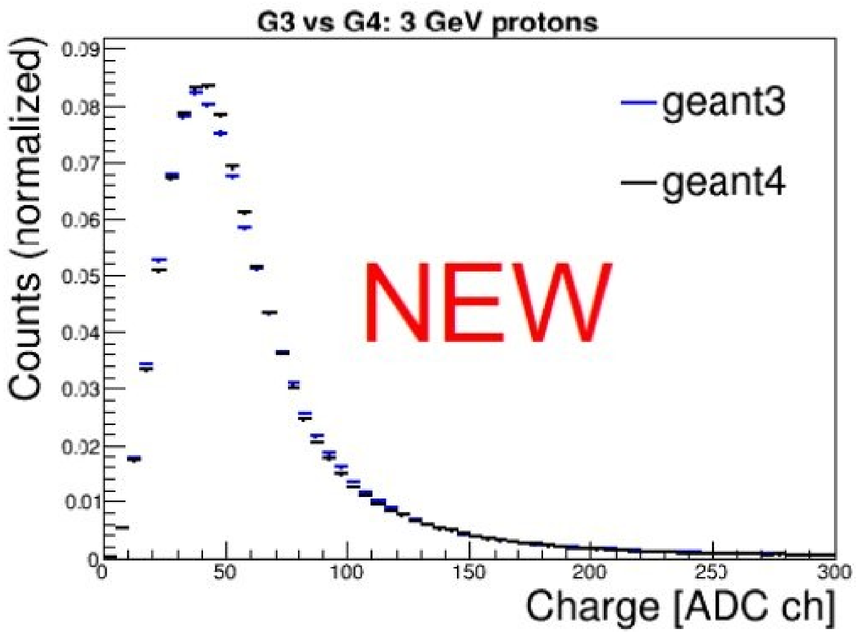
Packages available in Grid (FAQ)

Package name	Dependencies	Install
VO_ALICE@AliRoot::v5-01-Rev-09	VO_ALICE@ROOT::v5-30-03-1,VO_ALICE@GEANT3::v1-12-3	Linux, Mac
VO_ALICE@AliRoot::v5-01-Rev-32d	VO_ALICE@ROOT::v5-33-02b-1,VO_ALICE@GEANT3::v1-14-2	Linux, Mac
VO_ALICE@GEANT3::v1-15a-7		Linux, Mac
VO_ALICE@GEANT3::v1-15a-8		Linux, Mac
VO_ALICE@GEANT4::v9.6.p02		Linux, Mac
VO_ALICE@GEANT4_VMC::v2-14a	VO_ALICE@ROOT::v5-34-08,VO_ALICE@GEANT4::v9.6.p02	Linux, Mac

Geant4 Validation (2)

- In 2013 solved the problem of the TPC response in G4 simulations
  - G3 uses a special ALICE/NA49 model which describes well the test beam data
  - The solution was to add additional fluctuations in the step where energy loss is converted to ionization using a tuned Gamma distribution on the ALICE side.





Profiling and Performance Optimization

- By Sandro Wenzel, CERN PH-SFT, in 2013
- Large performance gap, factor 3 between Geant3 and Geant4
- AliRoot benchmark (pp@7TeV)
  - Profiling with Geant4 9.6.p01 with Valgrind, igprof, Intel PIN tools
  - SLC5, SLC6 with gcc 4.3.6, gcc 4.7
- Performance penalty
  - Due to use of G4EnhancedVecAllocator
  - Due to a small step limit was imposed in low density materials
- Obtained gain in performance:
  - 40% improvement (7400 s -> 4430 s time for 50 events)

Profiling and Performance Optimization (2)

- AliTransportMonitor tool developed by A. Gheata to identify and understand main consumers, hot spots, G4/G3 differences
  - Transport time, number of steps, mean energy, is scored per volume and particle type
  - Output (array of AliTransportMonitorVol objects) can be analyzed offline)
- Observations
  - Main consumer: ZDC sub-detector (G3: 50%, G4: 60%) - a candidate for fast simulation and/or simulate only if needed for physics analysis
  - Differences in G3/G4 time spent in the volume in most contributing volumes:
    - ALIC (ALICE container volume): G3: 43%, G4: 14%
    - TPC Drift volume: G3: 11%, G4: 16%
  - Not expected overall ~20% contribution in Beam pipe volumes

New Ways To Speed Up: Multi-Threading

- Development of Geant4 VMC MT started at end of 2011
- The first prototype was described at CHEP 2012 poster: “The Geant4 Virtual Monte Carlo”
  - Geant4 VMC code was adapted for multi-threading using the same approach as in Geant4 MT
  - Replacement of the singleton objects in Geant4 VMC with singletons per thread
    - Including the main classes: TVirtualMC and TVirtualMC application
  - Applied modifications, as described in the Geant4MT User’s Guide, to the Geant4 VMC classes, mainly in geometry and run categories
  - Adapting ROOT output to multi-threading as Geant4 VMC is closely integrated with ROOT
    - As Geant4 MT introduces parallelism per event, the ROOT output can be introduced per thread: each thread opens and writes on its own ROOT file
  - G4Root was not yet migrated in this prototype

New Ways To Speed Up: Multi-Threading (2)

- After Geant4 10.00 release, the Geant4 VMC prototype had to be adapted to changes in the Geant4 interfaces for user application
  - Straightforward as a recipe available in Geant4 examples
  - Utilities in G4Threading namespace

Geant4 VMC 3.00

- Geant4 VMC version which provides support for Geant4 multi-threading mode
  - First (beta) version released **on 14 March 2014**
- With participation of A. Gheata, CERN
  - Migration of G4ROOT
- Single source code for both sequential and multi-threading modes
  - VMC applications which will not be migrated for MT can be build and run with the same Geant4 VMC as migrated applications
- MT mode is activated automatically when Geant4 VMC is built against Geant4 MT libraries
- All (5) VMC examples were migrated to MT and can be run in this mode both with Geant4 native and Root navigation

Geant4 VMC 3.00 (2)

- New set of classes for ROOT IO management which take care of locking critical operations (registering ROOT object to trees etc.) is introduced in new **mtroot** package
  - http://root.cern.ch/drupal/content/mtroot
- The instructions for migration VMC applications to MT are available from VMC Web site:
  - http://root.cern.ch/drupal/content/multi-threaded-processing
- Besides MT, there are added VMC application main programs together with CMake configuration files which allow to run VMC without dynamic loading of libraries
  - A test suite was added in addition to the existing one for testing the examples from application programs

Home » VMC » Geant4 VMC

Multi-threaded processing

Geant4 VMC with Multi-threading Geant4

Since version 3.00 (to be released), Geant4 VMC supports running Geant4 in multi-threading (MT) mode. The VMC application will run automatically in MT mode when Geant4 VMC is built against Geant4 MT. VMC application which has not been migrated to MT should be built and run with Geant4 VMC built against Geant4 sequential libraries, but its build against Geant4 MT is not supported and its run will stop with an exception.

As the VMC classes work as a factory for creating Geant4 application objects (user initialization and user action classes, sensitive detector classes etc.), the main VMC objects: TGeant4 and MCApplication need to be created on both master and worker threads. Creating of all objects on worker threads is triggered from Geant4 VMC classes. Users need just to implement new functions of TVirtualMCApplication which are then used to clone the application and its containing objects on workers:

```
// required for running in MT
virtual TVirtualMCApplication* CloneForWorker() const;
// optional
virtual void InitForWorker() const;
virtual void BeginWorkerRun() const;
virtual void FinishWorkerRun() const;
virtual void Merge(TVirtualMCApplication* localMCApplication);
```

Overriding of TVirtualMCApplication::CloneForWorker() is required, implementation of the other functions is optional.

Remaining Issues

- Geant4:
  - Very frequent breaks when running with optical processes in multi-threading mode
    - Open Geant4 bug report #1590
  - Other minor issues to be reported
- ROOT
  - A fix proposed in TCint class to avoid a dead-lock when running test programs with calling ROOT macros, the fix should be included in the next ROOT 5.34.x iteration
  - Other minor issues to be reported

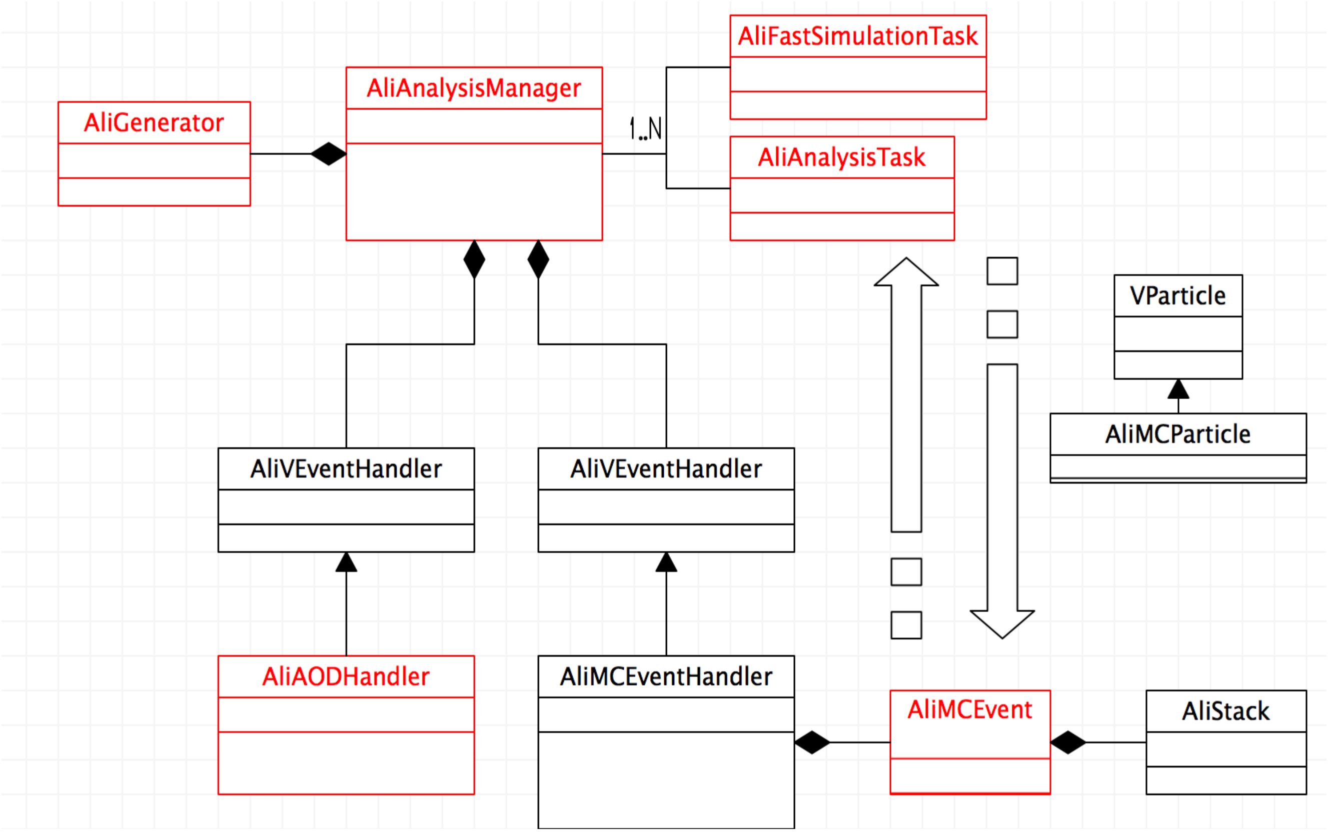
Others Ways To Speed Up

- Besides migration to Geant4 multi-threading, continue to look at possible improvements of the actual setup
- Linking the VMC application with static libraries instead of dynamic loading of libraries
  - Speed up ~ 10% in sequential mode and > 20% in multi-threading mode observed in VMC tests
- Revision of “stack popper” process (which adds user defined particles in tracking during event processing)
- Revision of cuts defined in Geant3 way
- Any other ideas ?

Options for Fast Simulation Integration

- Parametric fast simulation (PFS)
  - Analysis: tries to invert effects of efficiency and resolution
  - Parametric fast simulation: applies effects of efficiency and resolution
- Natural place for PFS: analysis framework
- Event generation (on-the-fly) has been already integrated.
  - reduced I/O speeds up analysis
- Analysis task applies efficiency and smearing

Kinematics Level Analysis + PFS



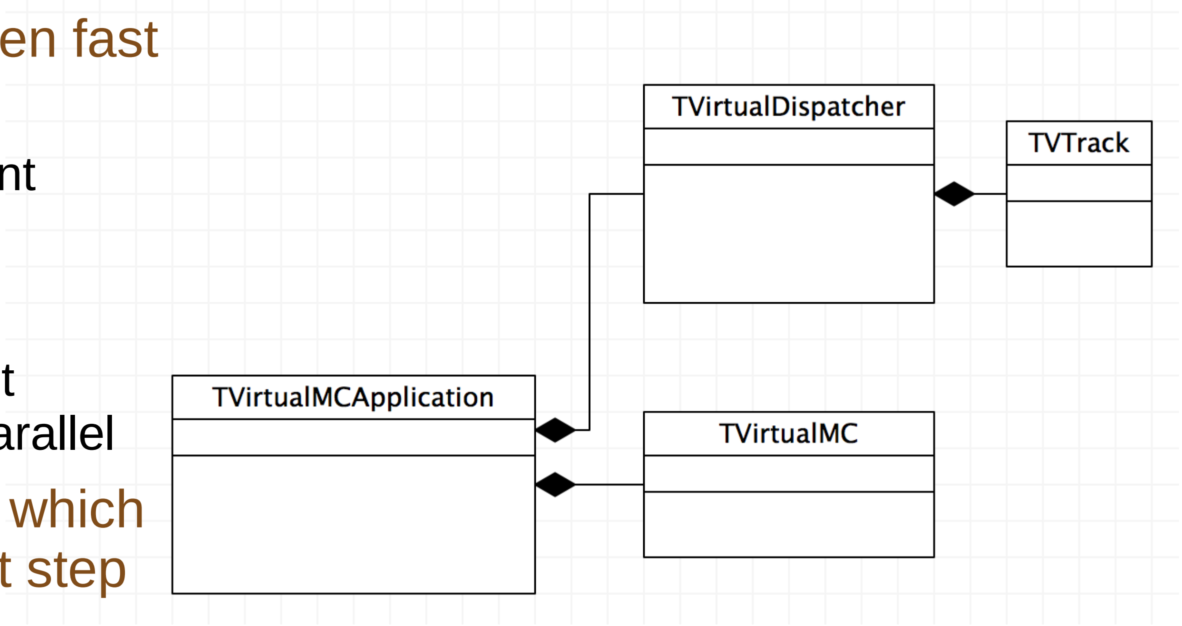
Write Physics Objects AOD

- Possibility to update MCEvent
- Possibility to chain with slow simulation transport
- AODs from fast and slow simulation have to be merged

Limited to primary particles

Options for Fast Simulation Integration (cont.)

- Full Fast Simulation (FFS)
  - Ideally transparent to user code
  - Possibility to switch between fast and slow simulation for
    - different particles, different detectors energies
  - Integration with VMC
    - Possibility to use different VirtualMC instances in parallel
  - Dispatcher object decides which instance performs the next step



- Geant4 fast simulation framework integration in Geant4 VMC
  - Explore Gflash Geant4 example

Plans

- Physics validation
  - Setup central GRID Geant4 productions and evaluate physics results (observables from electromagnetic processes: secondary electrons, optical photons, ...)
- Tests with multi-threading
  - Short term (within few months):
    - Performance tests with Geant4 VMC 3.00 + ALICE geometry
  - Long term (within one/two year(s)):
    - Tests with AliRoot with a limited detector setup (requires migration of AliRoot VMC application, AliMC for MT)
    - Follow new AliRoot 6 framework
- Fast Simulation framework

Conclusions

- ALICE simulation is based on Root tools:
  - Virtual Monte Carlo, Root geometry modeller TGeo and G4Root navigation interface
- Geant3 is used as the production Monte Carlo
- Geant4 is considered as a replacement for productions for Run 2
  - Physics results validation done in 2011
  - Need for speed up: hope to gain it with migration to multi-threading
- Fast simulations framework
  - Replacement for the critical parts of simulations