CBM (Fair) Simulation & Analysis Framework
CBM(Fair)Root

Mohammad Al-Turany (IT-GSI)
Denis Bertini (IT-GSI)
Florian Uhlig (CBM/IT-GSI)
Ilse König (Hades-GSI)

Motivation

- Start in 2003 as simulation package for CBM (CbmRoot)
  - Work fast! (LOI, TSR deadlines)
  - Make reliable simulation in short time
  - Which simulation engine to choose?
    - One would like to use the modern and maintained GEANT4
      - lack of knowledge about GEANT4 (intrinsic cuts / physics list …)
      - It was extremely difficult to get support for working with Geant4
    - Better knowledge of “old” MC’s: GEANT3, FLUKA
  - Limited manpower
    - Reuse as much as possible of already existing code
    - Profit by developments done for ALICE (VMC, EVE, AliEn)

General Requirements:

The software should be:

- Easy to use specially for:
  - Students that are not so experienced in detector simulation
  - Physicists familiar with Geant3
  - Non professional programmers
- Easy to maintain (Re-use of existing standard tools):
  - Use ROOT as a base framework
  - Reuse some functionality of the HADES Analysis Framework Hydra (*Geometry interface, Parameter Handling and Oracle interface*)
- Easy to install
  - Don't depend on special UNIX flavor and compiler
    - Support most Linux flavors in the community (MAC OS X 10.4 and 10.5 are supported now, Sun Solaris is work in progress)
    - Supported compilers, gcc, icc and cc
- Automated quality assurance is a must (Dashboard, CDash, Rule checker)

Guide lines for developments: (1)

1. Re-use existing and standard packages if they do the job
   - Autoconf/automake (later CMake)
   - Doxygen for documentation
   - CVS for code management (later SVN)
   - Oracle for the parameters and geometry

2. The code is released immediately to the developers via CVS/SVN repository. (fast feed back about bugs or missing features)

3. Two or three stable releases per year for users which only wants to do analysis.

4. The code should be modular and allow fast exchange of algorithms or detector code

Guide lines for developments: (2)

4. Keep the framework as small as possible. Features are only implemented when they are really needed. Taking care that the interfaces are well defined so that new functionality can be easily added.

5. Performance and stability has the highest priority. They are more important than idealistic views of object oriented or design concepts which may look much nicer but kill the performance or stability!
   - No STL in Trees, but TClonesArray
   - C- Array for the Field Maps

CbmRoot

- In very close contact with CBM collaboration implementing a VMC based framework was started, the starting points where:
  - ROOT based.
  - Use basic ideas and structure of AliRoot
  - Re-Use of Hades geometry interface
  - Re-Use of Hades Runtime database and Oracle interface
  - TGeoManger as a navigation from the beginning
  - Extended TTask for reconstruction and analysis

Main Features:

- No Executable:
  - Compiled Tasks for reconstruction, analysis, etc.
  - Root macros for steering simulation, reconstruction and analysis
  - Root macros for configurations (G3, G4, Fluka and Analysis)

  Root is the executable which runs the steering macros, thus the same macros runs on a local PC, the batch farm or even on the grid!

- VMC and VGM for simulation:
  - Running different transport MC's from the same application
  - No need to customize detector response, physics configuration and cuts (partially) or input/output according the transport code
  - Geometry is described once and then one can choose between different MC's and different navigations: e.g:
    - G3 Native geometry and navigation
    - G4 Native geometry and navigation
    - G4 Native geometry and Root navigation
    - G4 Root geometry and navigation
    - .....

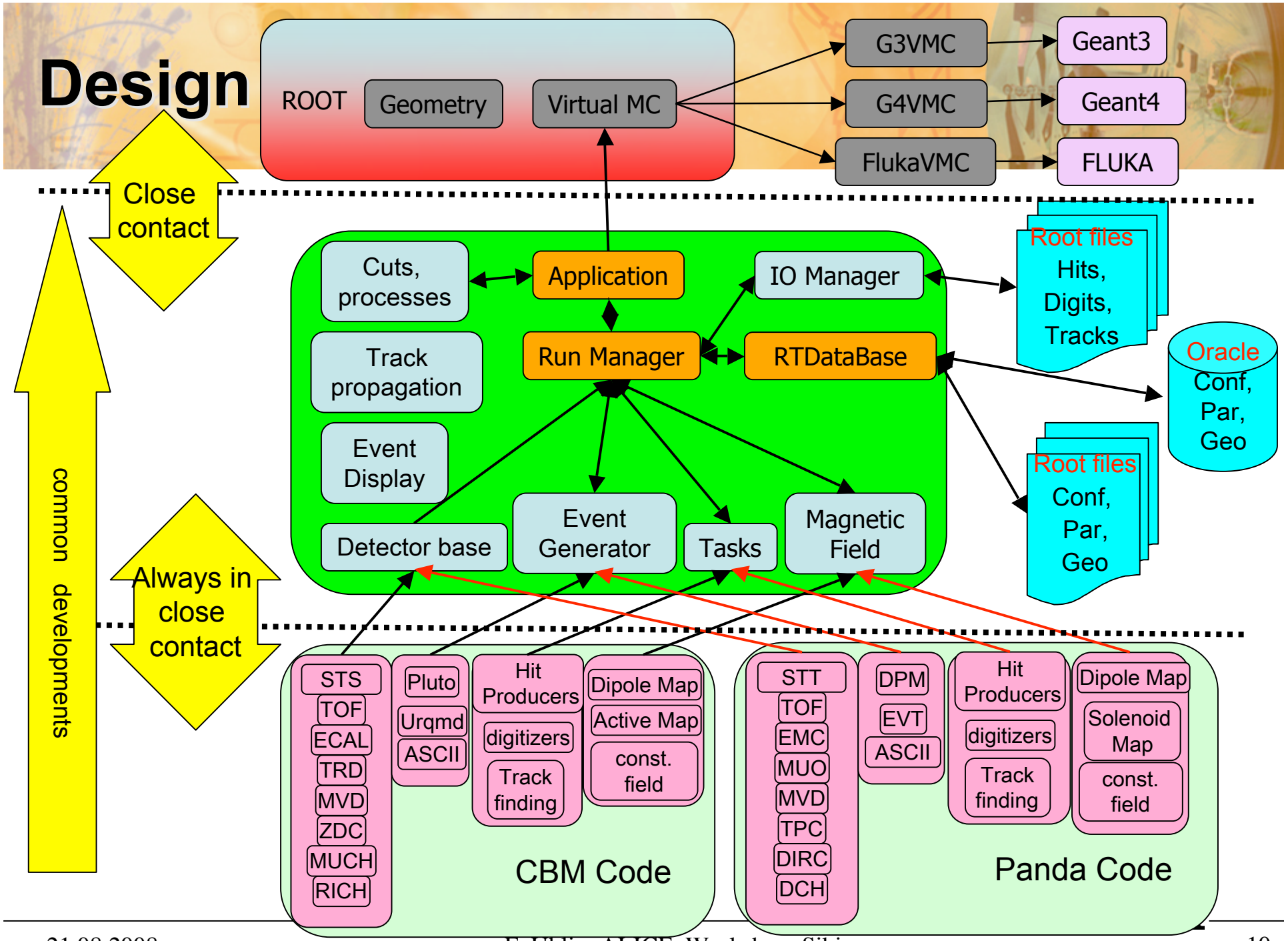
Main Features: (2)

- Dynamic Event structure:
  - Detectors can be added or replaced on the fly without changing the compiled code
  - For design studies it is easier to work with a subset of the detectors and then easily merge the rest
  - Simple analysis on output files can be done in plain root without any extra libraries
  - Connection of different files can be done via tree friends without copying any information between different data levels
  - Transient and persistence objects are the same
- Fast Simulation
  - Event generators just push the event into the stack, no transport is taking place
  - Detector response is presented as TTask
  - The output has the same format as full simulation

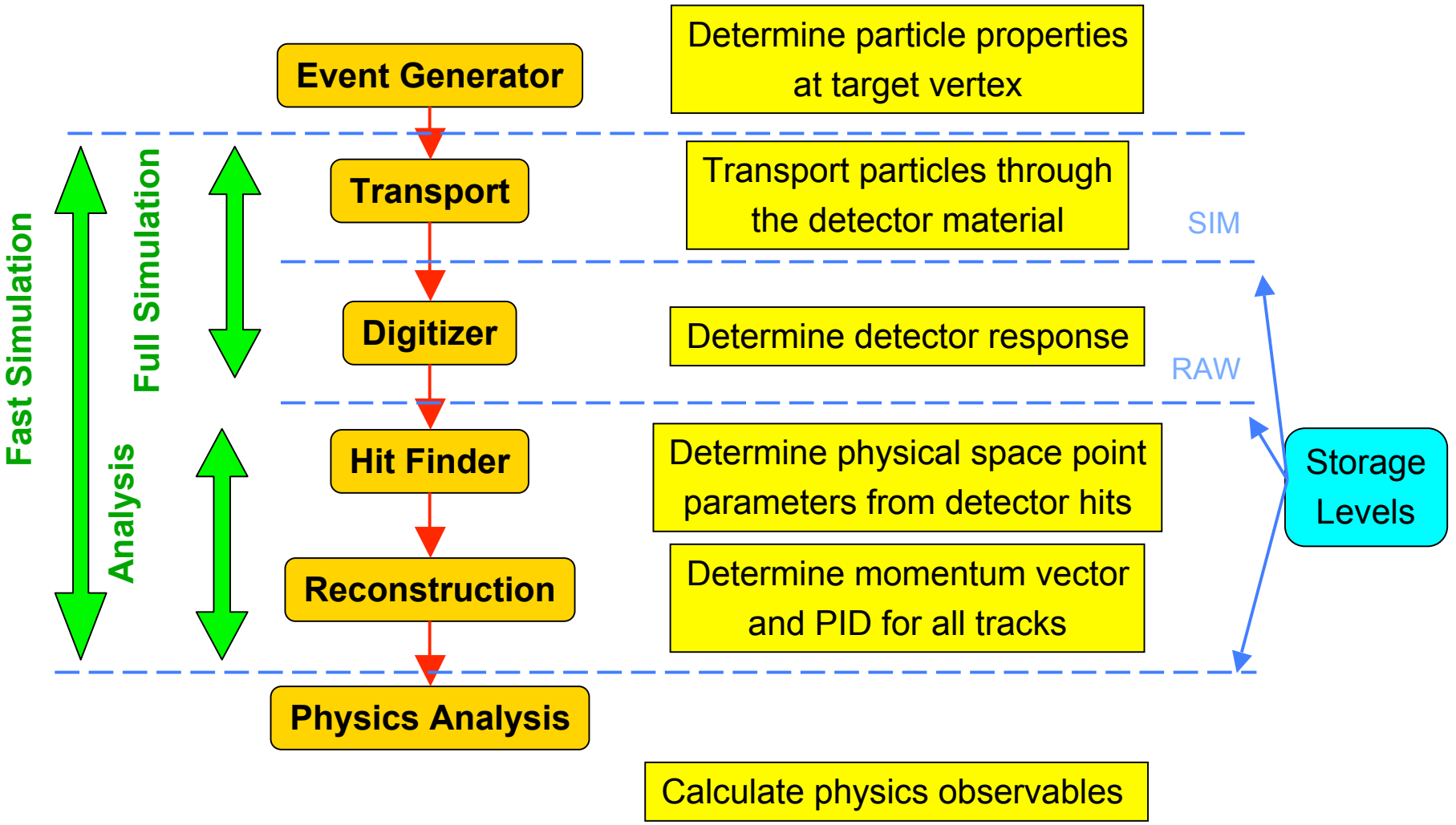
Status

- In 2006/2007 the Panda Experiment decided to use the base classes of CbmRoot
  - Decouple base classes and rename project to FairRoot
  - Base classes are used by both experiments
  - Panda was able to setup a productive environment in very short time
  - Move from CVS to SVN
  - Move from autotools to Cmake
- The fact that FairRoot is used by many experiments meanwhile require more work on the different requirements. But also these experiments are contributing very strongly to the developments.
- Collaborating with ALICE, ROOT and VMC teams is still ongoing, prove to be very useful.

Design


ROOT
Geometry
Virtual MC
G3VMC
G4VMC
FlukaVMC
Geant3
Geant4
FLUKA
Close contact
common developments
Always in close contact
Cuts, processes
Application
IO Manager
Track propagation
Run Manager
RTDataBase
Event Display
Detector base
Event Generator
Tasks
Magnetic Field
Root files
Hits, Digits, Tracks
Oracle
Conf, Par, Geo
Root files
Conf, Par, Geo
STS
TOF
ECAL
TRD
MVD
ZDC
MUCH
RICH
Pluto
Urqmd
ASCII
Hit Producers
digitizers
Track finding
Dipole Map
Active Map
const. field
CBM Code
STT
TOF
EMC
MUO
MVD
TPC
DIRC
DCH
DPM
EVT
ASCII
Hit Producers
digitizers
Track finding
Dipole Map
Solenoid Map
const. field
Panda Code

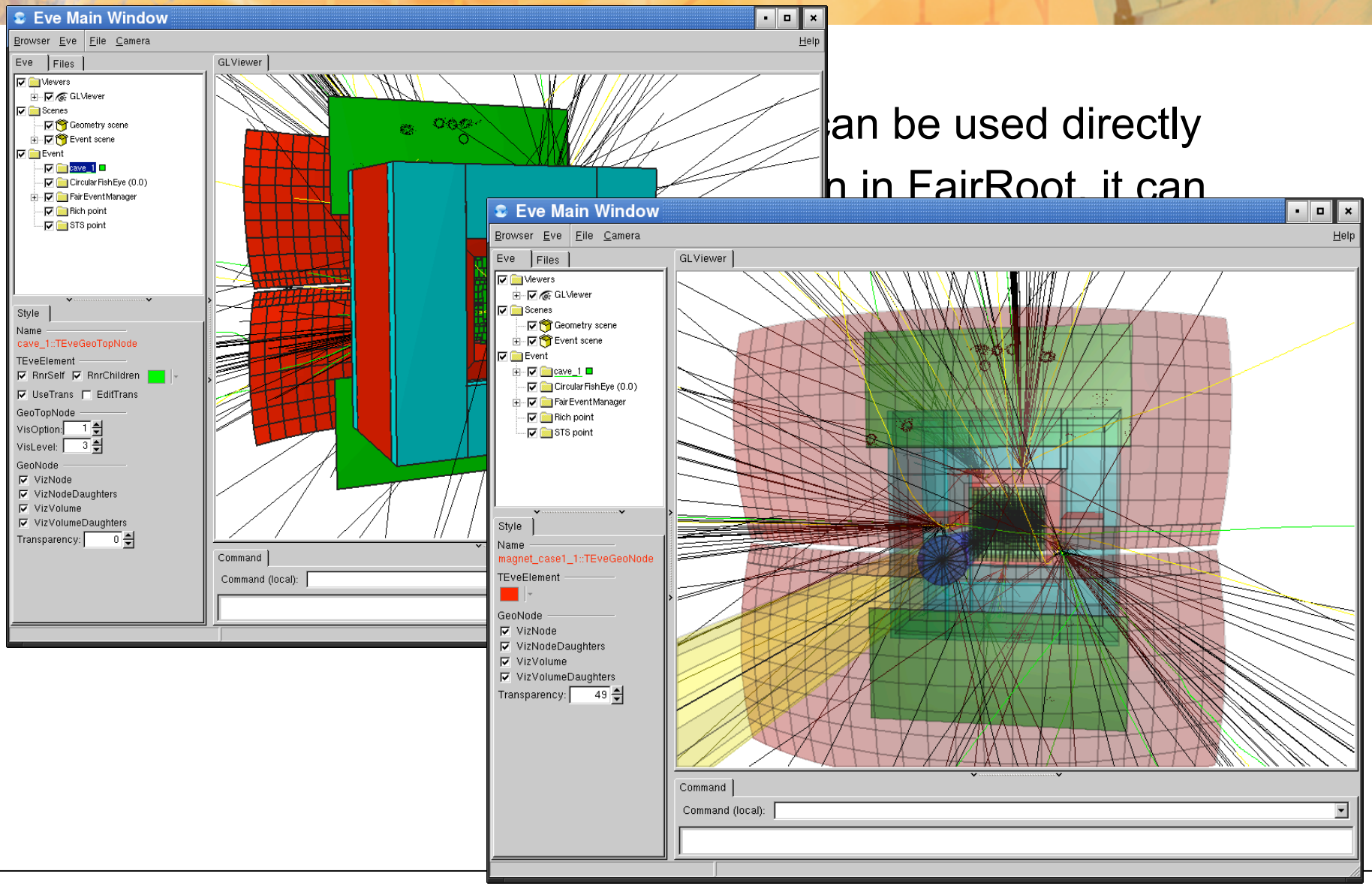
Simulation-Reconstruction Chain



Eve in FairRoot

an be used directly

n in FairRoot, it can



CMake

- Why Cmake:
  - It supports great variety of platforms
    - (basically every *ix, Mac OS, Windows)
    - (Win: Borland, MS Visual C++, cygwin, mingw)
  - CMake generates out of simple rules native Makefiles for all supported platforms
  - produce project files for IDE's (KDevelop, XCode, VStudio)
  - Input files (rules) are the same on all platforms
  - Big community behind it, CMake is the build tool for KDE 4
  - CMake has a testing framework

Why test daily

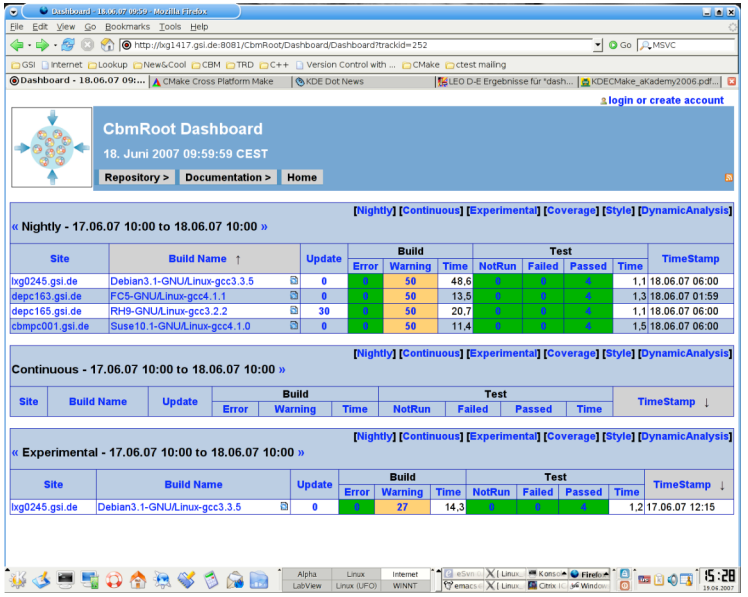
- Large code base is too large/complicated for a single developer to understand/maintain. It becomes even more complicated when dealing with several platforms.
- Identify problems when they occur.
- Project depends on external packages which can cause problems.
- Provide direct feedback to the developers as they experiment with new features.

Automtic testing of repository

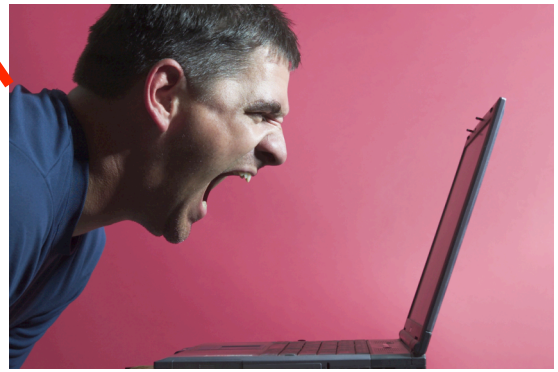
SVN maintains source code revision

CTest/CMake compiles and test the newly commited source code on distributed clients

http://fairroot.gsi.de/CDash



Typical developer checks in code

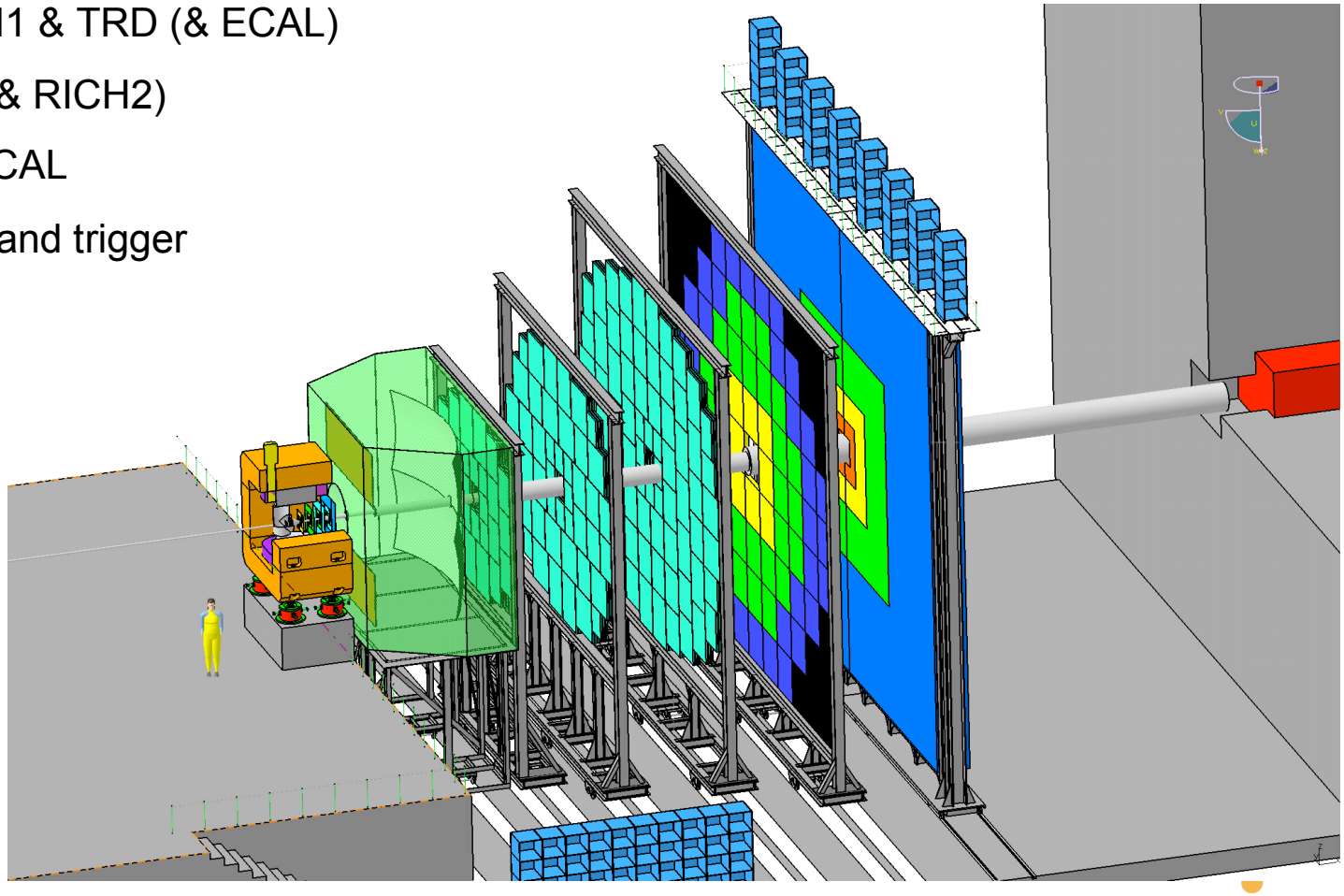


Developer reviews the results

FairRoot is used at:

CBM experiment@GSI

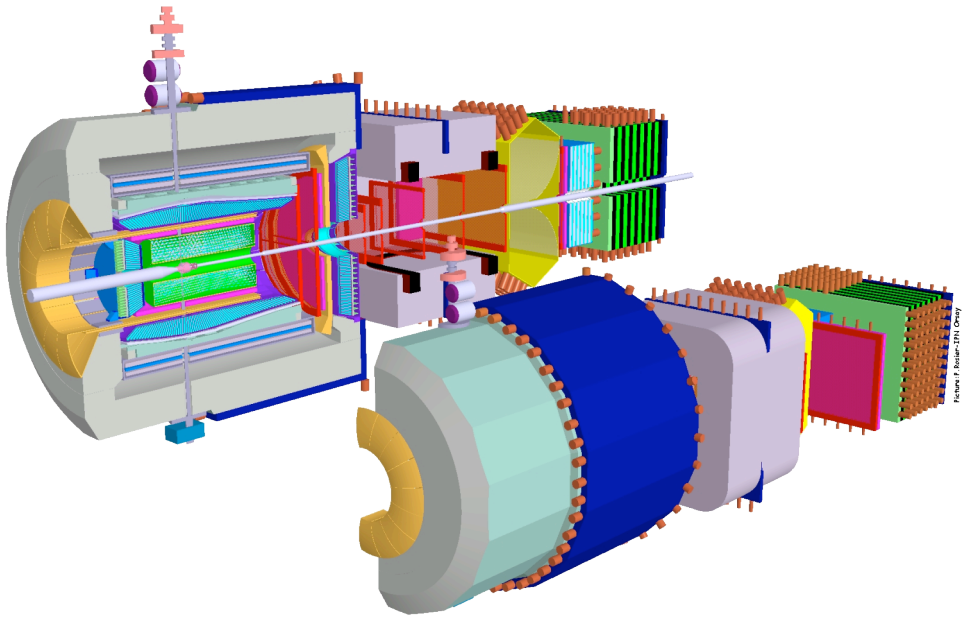
- tracking, vertex reconstruction: radiation hard silicon pixel/strip detectors (STS) in a magnetic dipole field
- electron ID: RICH1 & TRD (& ECAL)
- hadron ID: TOF (& RICH2)
- photons, π^0, μ: ECAL
- high speed DAQ and trigger



Panda Experiment@GSI

Multi purpose detector at FAIR

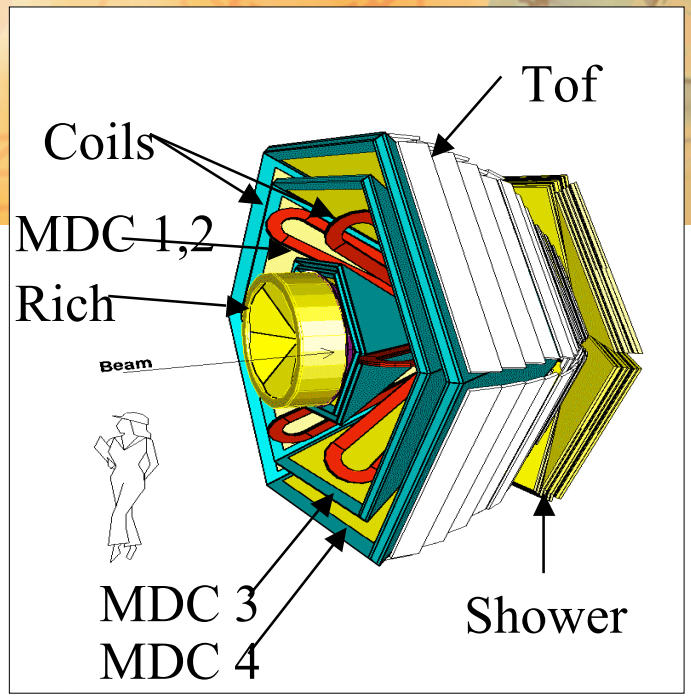
$\bar{p}p$, $\bar{p}A$ collisions
1.5⇒15 GeV/c ($\bar{p}$ momentum)



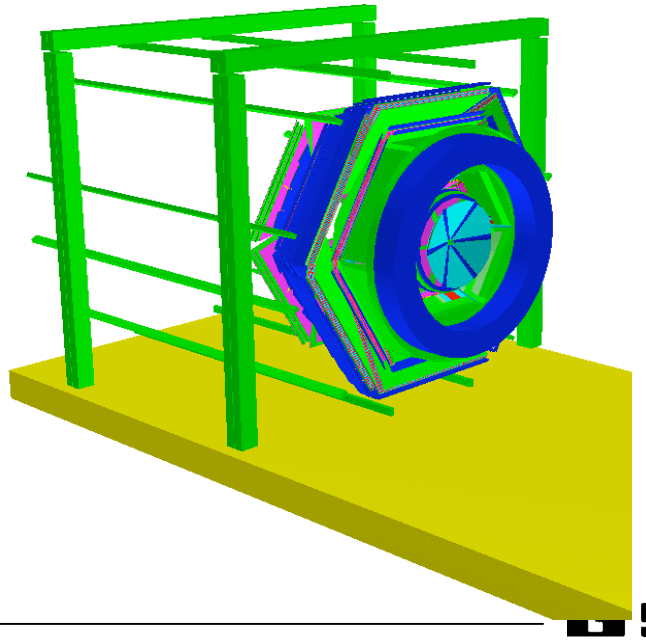
Physics program

- Charmonium ($\bar{c}c$) spectroscopy
- Open charm spectroscopy
- Search for gluonic excitations (hybrids - glueballs)
- Charmed hadrons in nuclei
- Single and double Hypernuclei
- Other options (EFF, GPD, …)

Hades@GSI

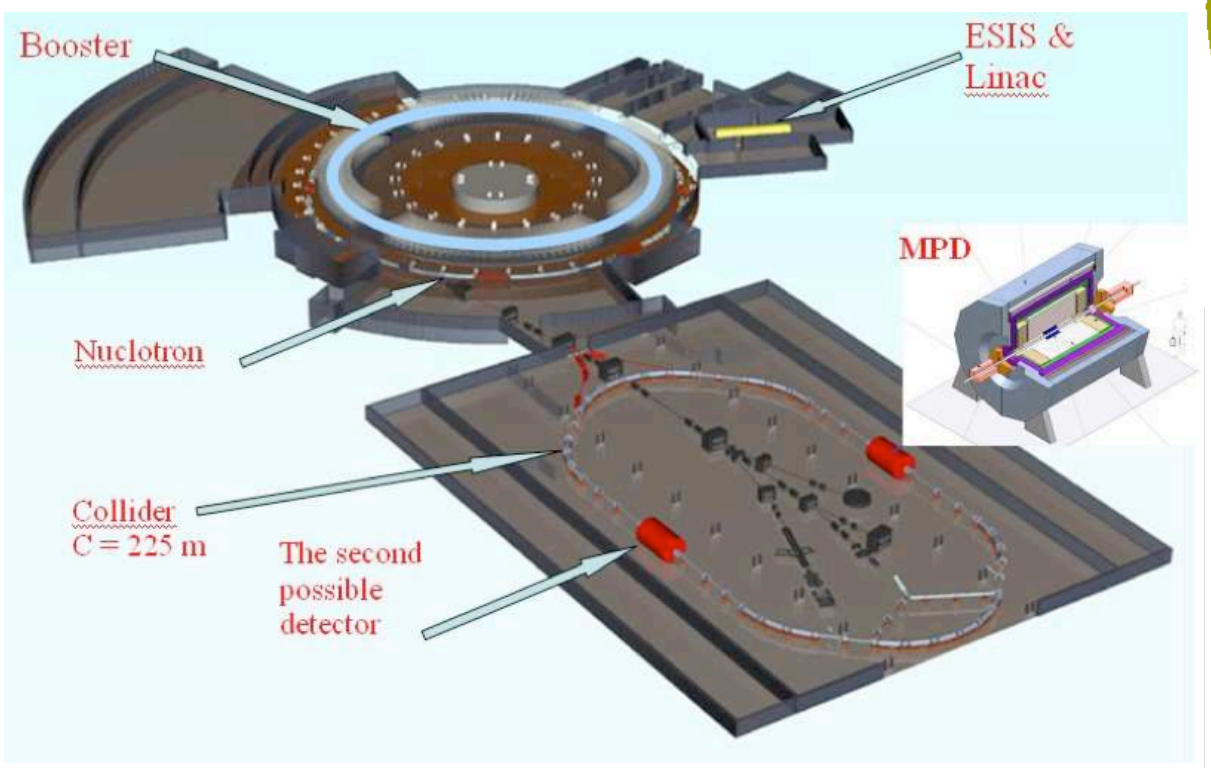


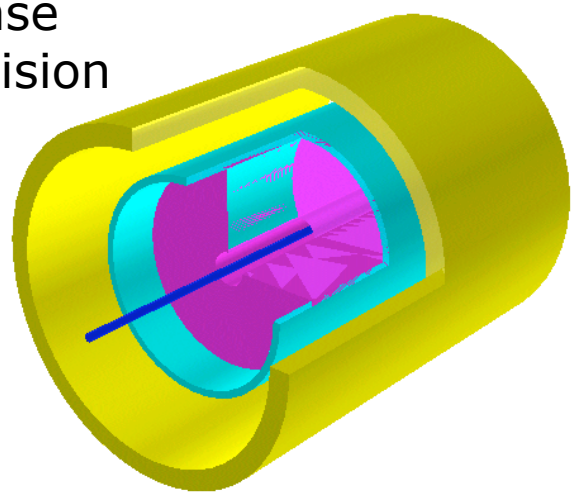
- **Goal:** Study of in-medium modifications (ρ, ω, φ) properties
  - Produced in A+A, p+A, π+A collisions
  - Di-electrons are used as probes: $V \rightarrow e^+e^-$
- Hexagonal symmetry around the beam axis
- Geometrical acceptance of 40%
- Invariant mass resolution of 1%
- Operates at reaction rates up to 10^6 /s
  $\Rightarrow$ 0.5 - 1 Tbyte/year
- ~ 70.000 readout channels

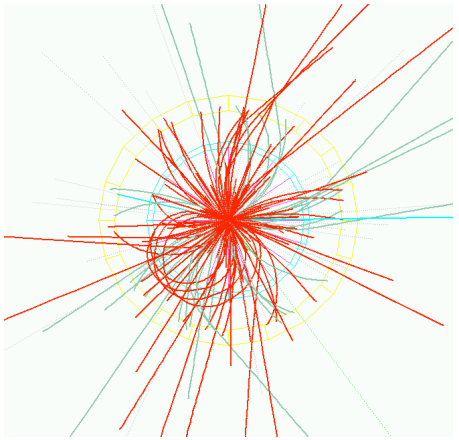


The Multi-Purpose Detector (MPD) at JINR (Dubna).

The NICA/MPD is aimed to study the hot and dense strongly interacting matter. It will be done in collision of heavy ions (up to the gold) at NICA collider



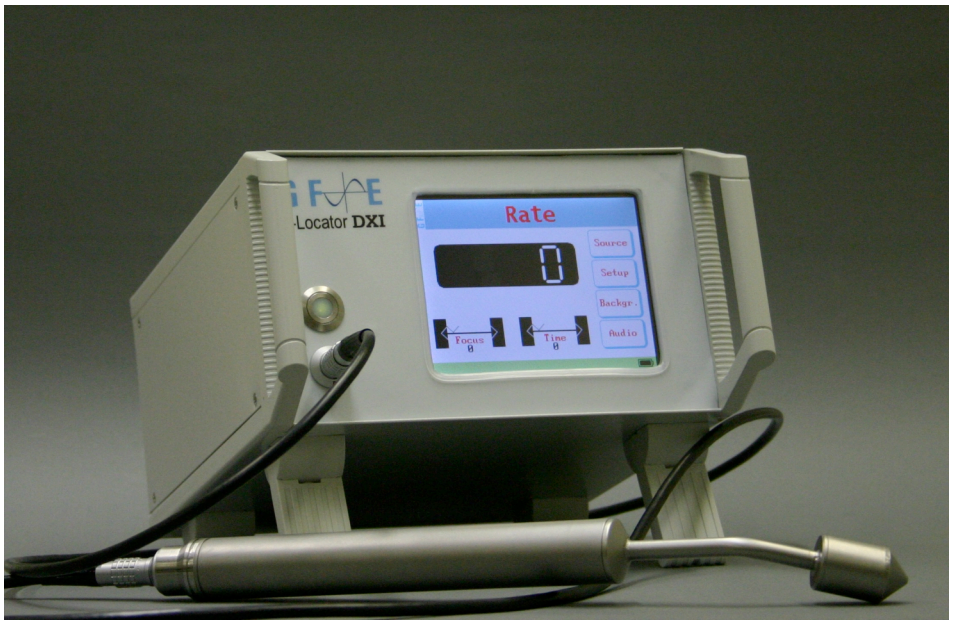


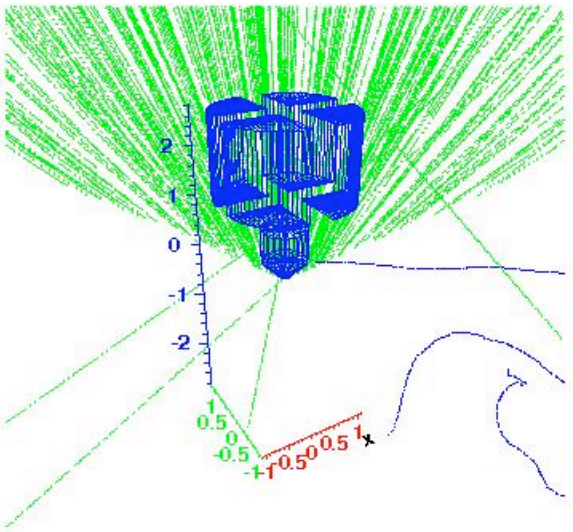


FairRoot@GF+E

Gesellschaft für Forschungs- und Entwicklungsservice mbH

(Gamma detector development for cancer therapy project at the GSI)





FairRoot used for detector optimization studies.

Summary and Outlook

- The approach to use Open Source software was very successful
- Very productive communication between the developers

- So far no changes in basic structure were needed to implement new features
- FairRoot is well established and used by several experiments
- Use synergy effects

- To test software automatically on different systems is a must

- Investigate multithreading support
- Investigate usage of GPU