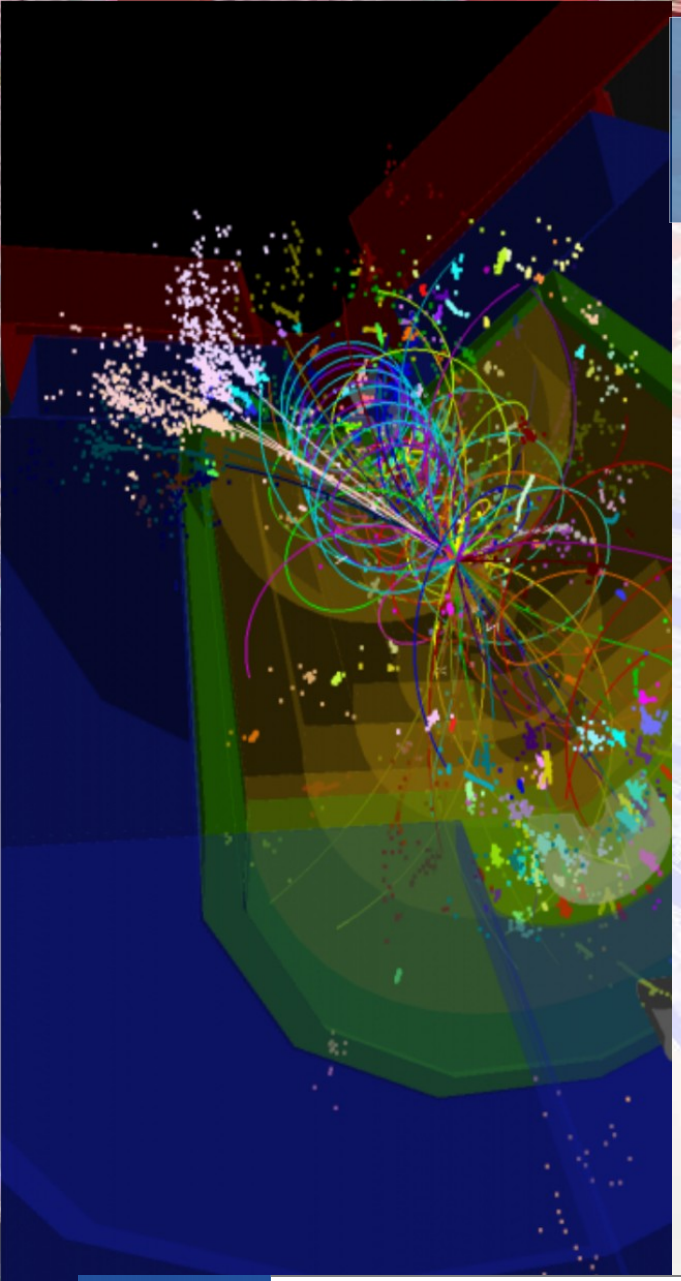


DD4hep

Conditions and Alignment Support for the DD4hep Detector Description Toolkit

M.Frank, F.Gaede, M.Petric, A.Sailer

This project has received funding from the European Union's Horizon 2020 Research and Innovation programme under Grant Agreement no. 654168.

AIDA 2020

Motivation and Goal

DD4hep

- **Develop a detector description**
  - **For the full experiment life cycle**
    - **Detector concept development, optimization**
    - **Detector construction and operation**
    - **"Anticipate the unforeseen"**
  - **Consistent description, with single source, which supports**
    - **Simulation, reconstruction, analysis**
  - **Full description, including**
    - **Geometry, readout, alignment, calibration etc.**

This project has received funding from the European Union's Horizon 2020 Research and Innovation programme under Grant Agreement no. 654168.

AIDA 2020

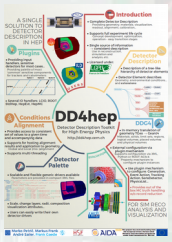
Saga in 5 Episodes: Sub-packages

DD4hep

- DD4hep – basics/core [1,2]
- DDG4 – Simulation using Geant4 [1,3]
- DDRec – Reconstruction supp.[4]
- DDCond – Detector conditions [5]
- DDAlign – Alignment support [5]

LGPL v3 Free Software

(1) Mature state: bug-fixes and maintenance
(2) M.Frank et al, CHEP2014, Amsterdam, NL
(3) M.Frank et al, CHEP2015, Okinava, Japan
(4) A.Sailer et al, CHEP2017, San Francisco, CA
(5) New Modules: this presentation



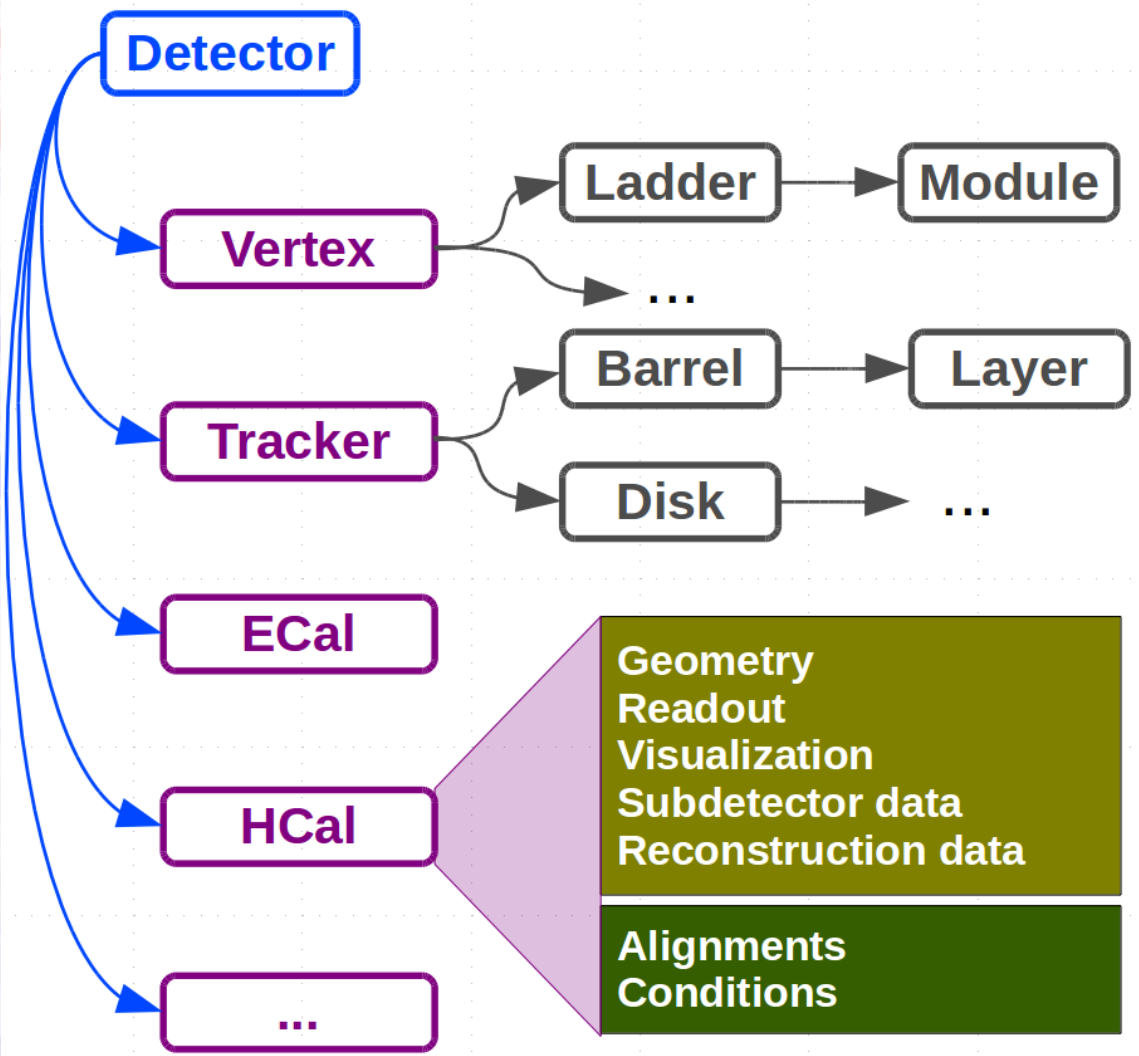
Poster contribution No. 2937617, 10 Jul 2018
New Developments in DD4hep

For further details see: http://dd4hep.cern.ch

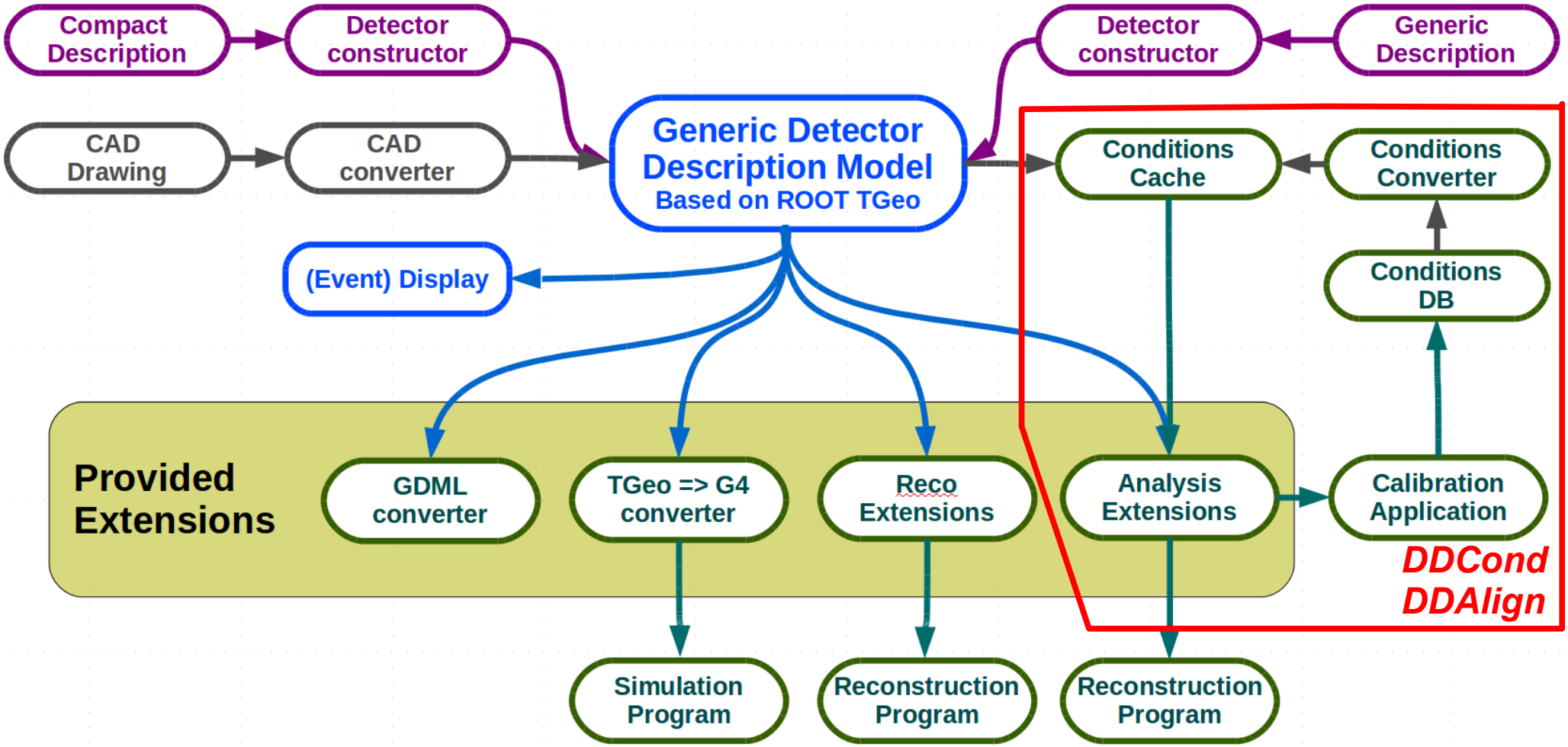
What is Detector Description ?

DD4hep

- Tree-like hierarchy of "detector elements"
  - Subdetectors or parts of subdetectors
- Detector Element
  - Structure + geometry
  - Contains optionally: subdetector or activity specific data
  - Provide access to time dependent data
    - Environmental data
    - Alignments
    - Derivatives of these



DD4Hep - The Big Picture



DDCond: Conditions Data

DD4hep

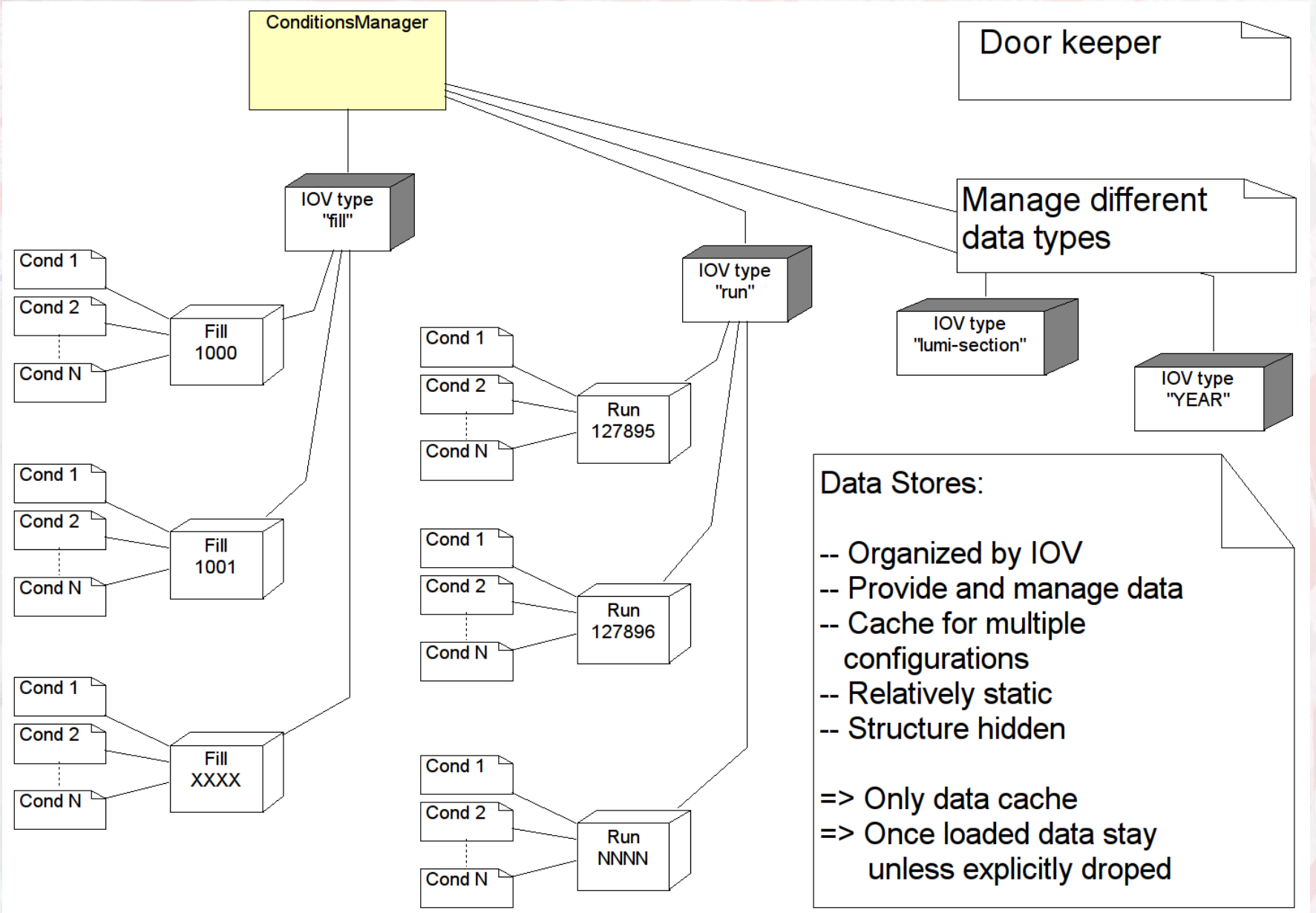
- **Time dependent data necessary to process the detector response [of particle collisions]**
  - **Slowly changing: every run O(1h), lumi-section O(10min) …**
  - **Conditions may be the result of computation(s)**
  - **Multiple conditions change in batches**
  - **Conscious design choice**
    requires design, compromise and discipline

- **DDCond deals with the conditions data**
  - **Efficient and fast resource management**
  - **Multi threading support by design**
    Well defined locking points
  - **Cache where necessary but not more**

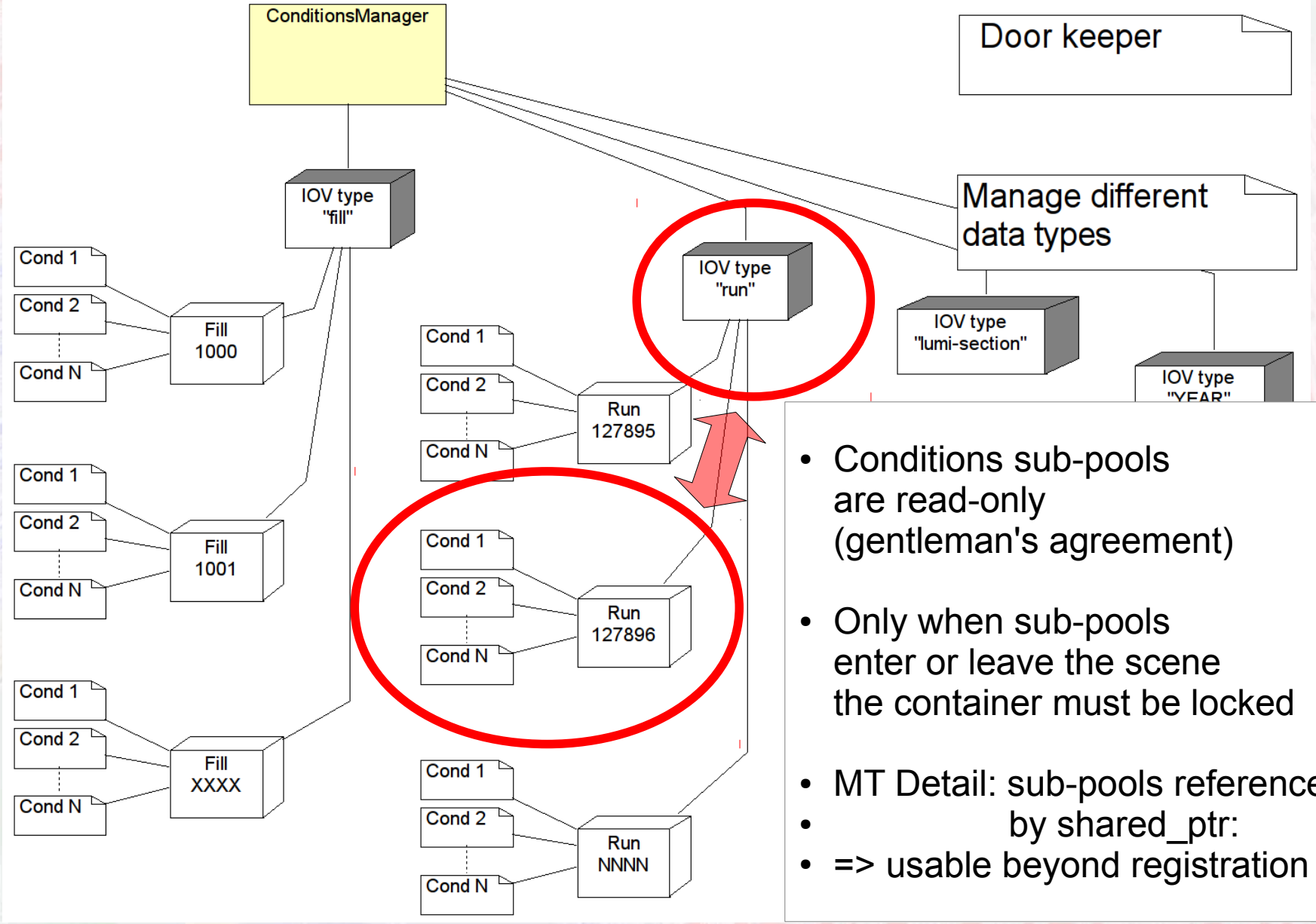
if used according to design ideas

DDCond Implementation
The Data Cache

DD4hep

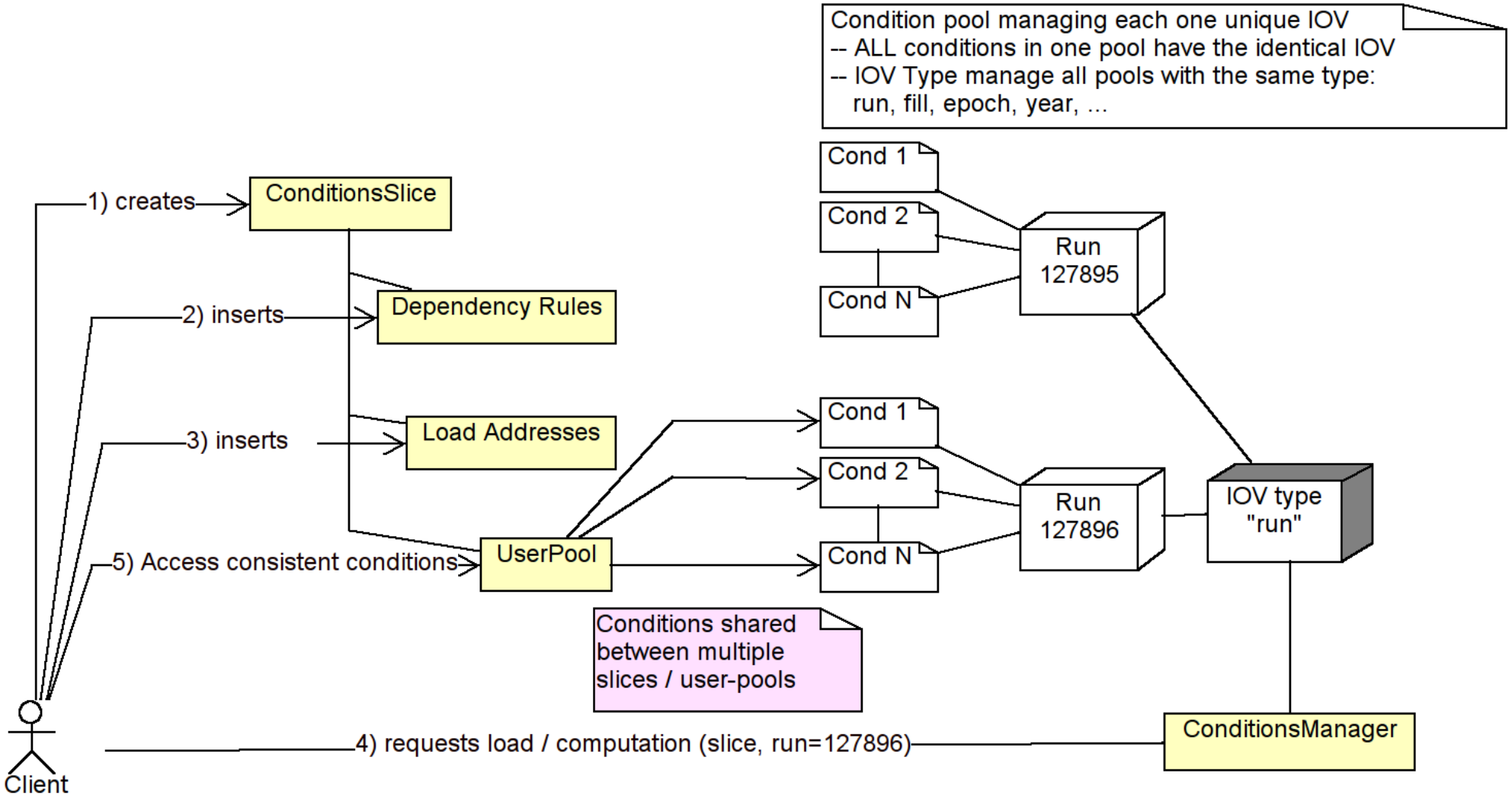


DDCond Implementation
The Data Cache



- Conditions sub-pools are read-only (gentleman's agreement)
- Only when sub-pools enter or leave the scene the container must be locked
- MT Detail: sub-pools referenced
- by shared_ptr:
- => usable beyond registration

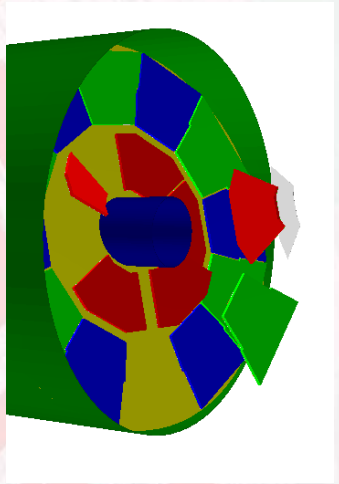
DDCond Implementation
IOV Slice Projection

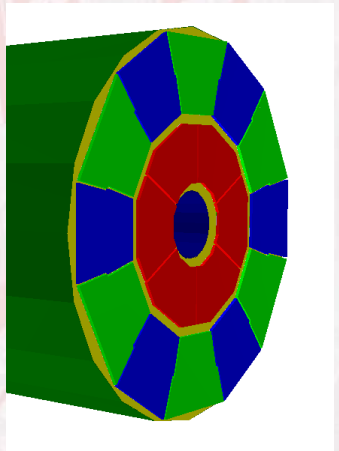


DDAlign: Global and Local Alignments

DD4hep

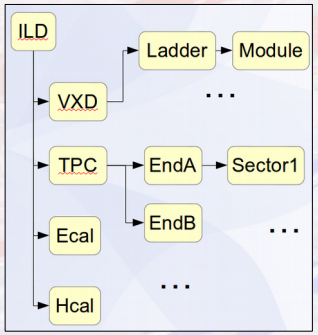
- **Global alignment corrections**
  - Physically alters geometry [unique to ROOT-TGeo]
  - By definition not multi-threaded
  - Possibility to simulate misaligned geometries
- **Local alignment corrections**
  - Geometry stays intact (either ideal or globally aligned)
  - Multi-threading supported, multiple versions
  - Local alignment corrections are conditions
    - Delta parameters: "raw" input data
    - Matrices from ideal geometry to world: derived data (e.g. adjust hit positions)

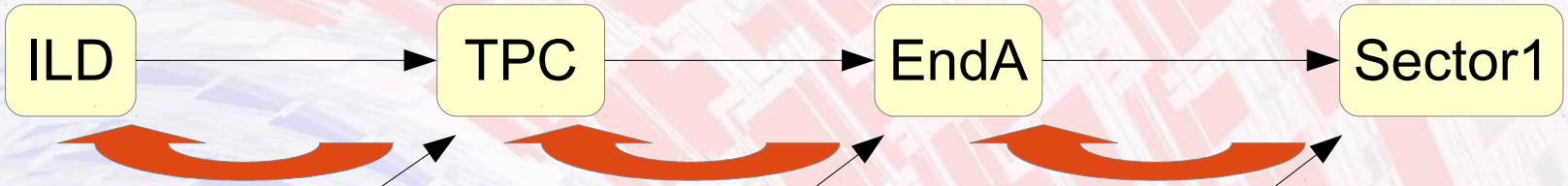




Local Alignment Δ - Parameters

DD4hep





$$Tr^{World}_{Sec\,1} = Tr^{World}_{EndA} \times \left(Tr^{Parent(EndA)}_{Sec\,1} + \Delta_{Sec\,1} \right)$$

$$Tr^{World}_{EndA} = Tr^{World}_{TPC} \times \left(Tr^{Parent(TPC)}_{EndA} + \Delta_{EndA} \right)$$

$$Tr^{World}_{TPC} = Tr^{World}_{ILD} \times \left(Tr^{Parent(ILD)}_{TPC} + \Delta_{TPC} \right)$$

- **Trickle-up the hierarchy and compute the matrices the most effective way with re-use of intermediate results**
- **Math verified by C. Burr**

DDCond and DDAlign: Status

DD4hep

- Described functionality was implemented and tested with LHCb conditions data
  - Persistent database from Run I, II (snapshot)
  - 9k conditions and 2.5k alignments
- Local Alignments handled as derived conditions
  - Convert Δ parameters (translation, rotation, pivot-point) to transformations to world or reference point
  - Implemented as C++ class `AlignmentsCalculator`
  - Used for alignment studies for the LHCb upgrade

Toolkit Users

DD4hep

Increasing interest in the HEP community

- ILC F. Gaede et al.
- CLICdp A. Sailer et al.
- SiD D. Protopopescu et al.
- FCC-eh P. Kostka et al.
- FCC-hh A. Salzburger et al.
- FCC-ee O. Viazlo (CLD design), N. Alipour, G. Voutsinas
- CMS Evaluation for upgrade started (202x) (Y.Osborne et al.)
- LHCb Evaluation for upgrade started (2019) (B.Couturier et al.)

  See also poster contribution 2937633, 10 Jul 2018
  "Perspectives for the migration of the LHCb geometry to the DD4hep toolkit"
- CALICE Calorimeter R&D, started
- EIC Evaluation considered/started (W. Armstrong et al.)

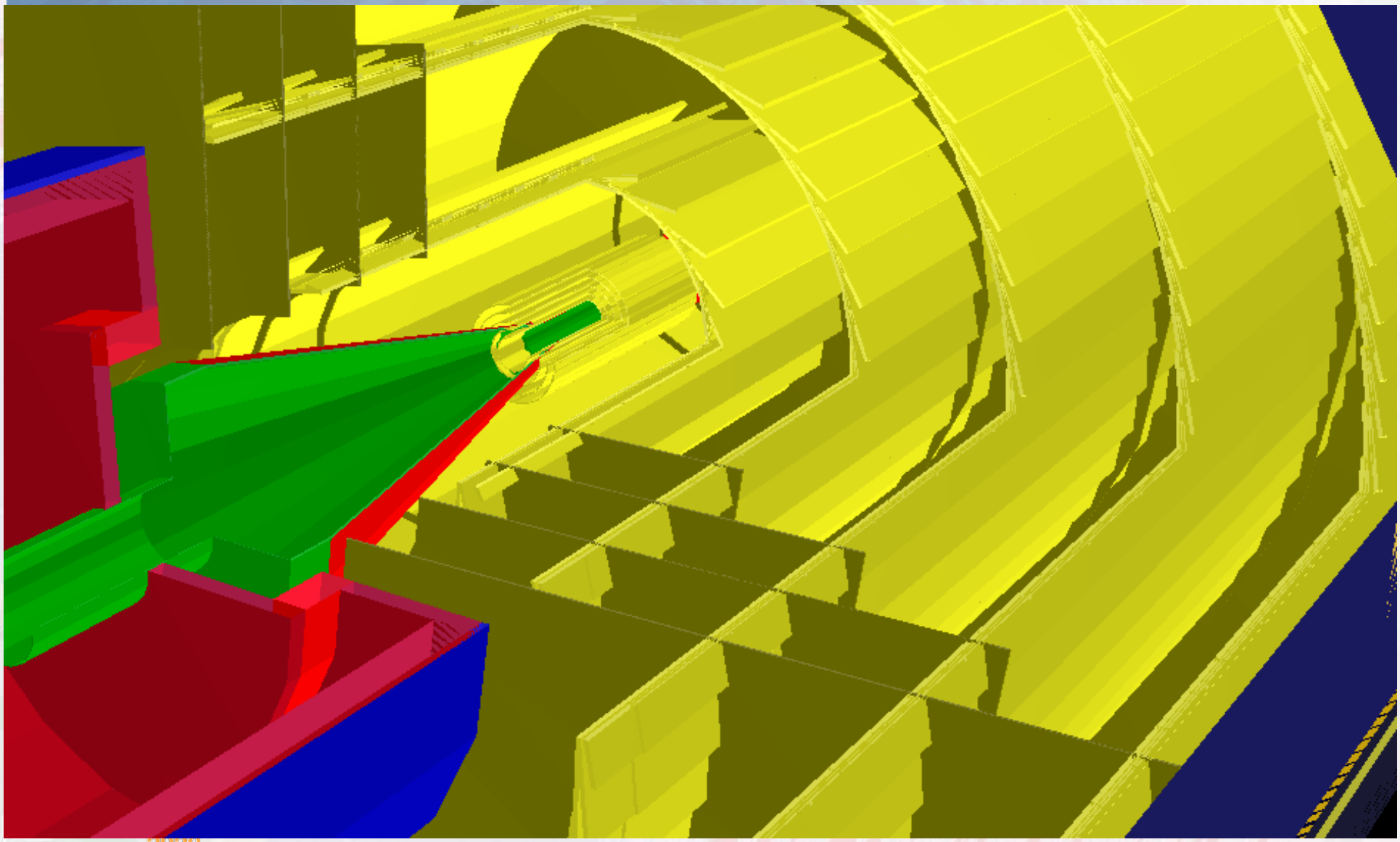
Summary

DD4hep

- DD4hep is mature now
- Handles all aspects of detector description for the lifetime of an experiment
- Access to conditions and alignments is implemented efficiently and supports multi-threading
- Increasing interest in the community and increasing number of users
- Visit us on:
  - `http://dd4hep.cern.ch`
  - Up to date Doxygen information
  - User manuals for further information have improved

DD4hep

Questions and Answers



Real World Use Case
LHCb Velo Detector

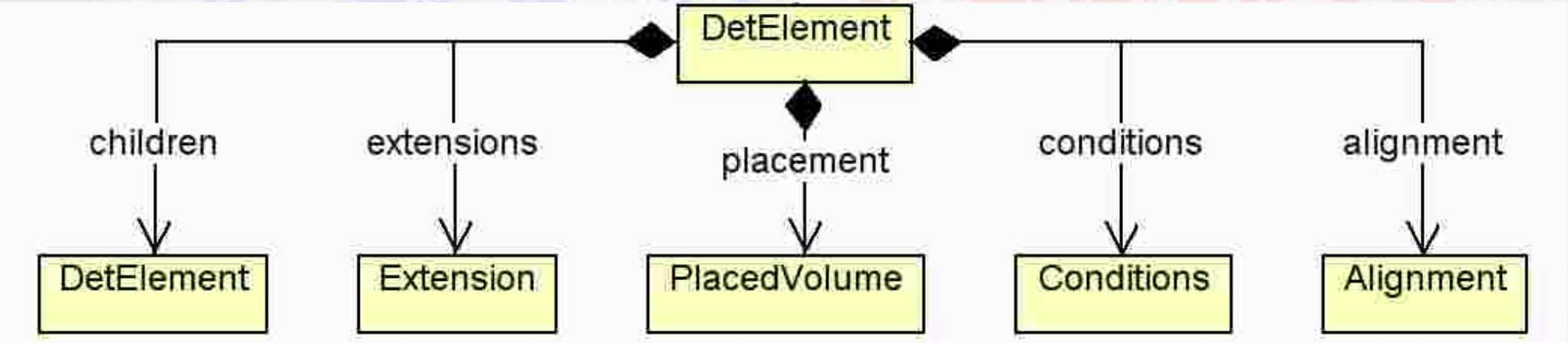
DD4hep

- Structures are build using the derived conditions callback mechanism
  - Static part: once only
  - IOV dependent part: when not in pools
    Also fills link to static information
- Since conditions in existing pools still can be shared while preparing new IOV depending conditions
  - No locking strategy necessary
- Alignment computation incorporated
  - Reminder: alignments must be computed ‘en block’
    for an efficient computation

Get Fingers Dirty
LHCb Velo Detector

DD4hep

- People want to see “Detector elements”
  - Fully functional description of parts of the detector
    - Long term valid stuff (structure)
    - Short lived quantities (temperature, alignment, …)
- “Natural” aggregation:

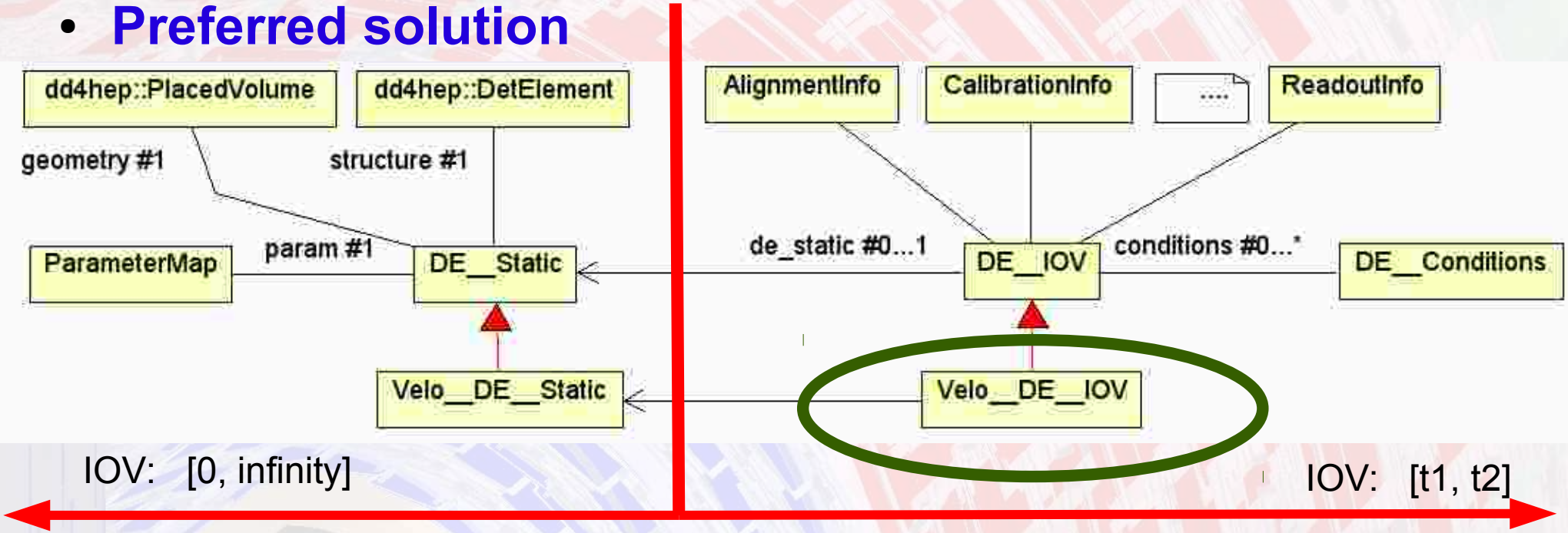


- Intuitive, but not good: no multi-threading

Real World Use Case
LHCb Velo Detector

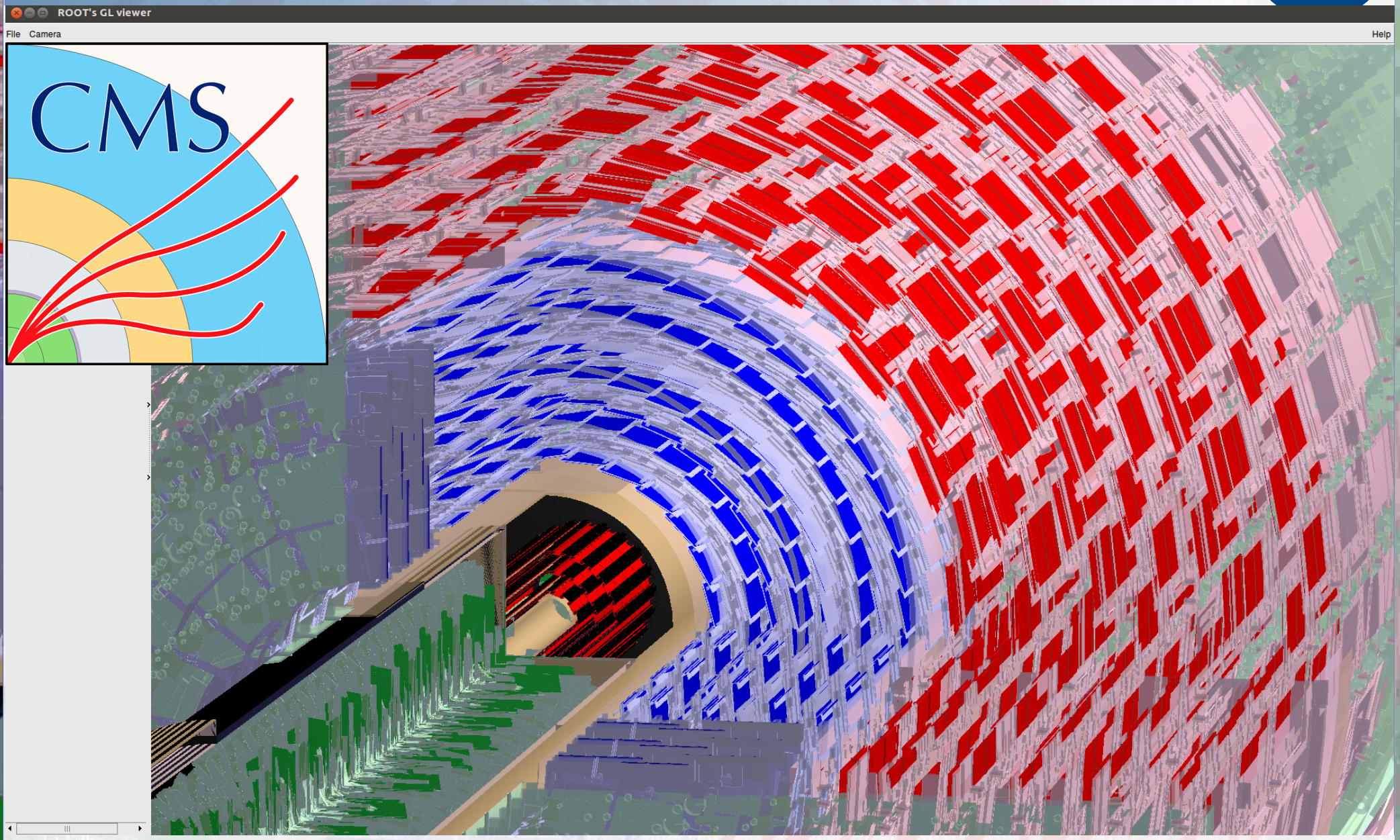
DD4hep

- **Preferred solution**



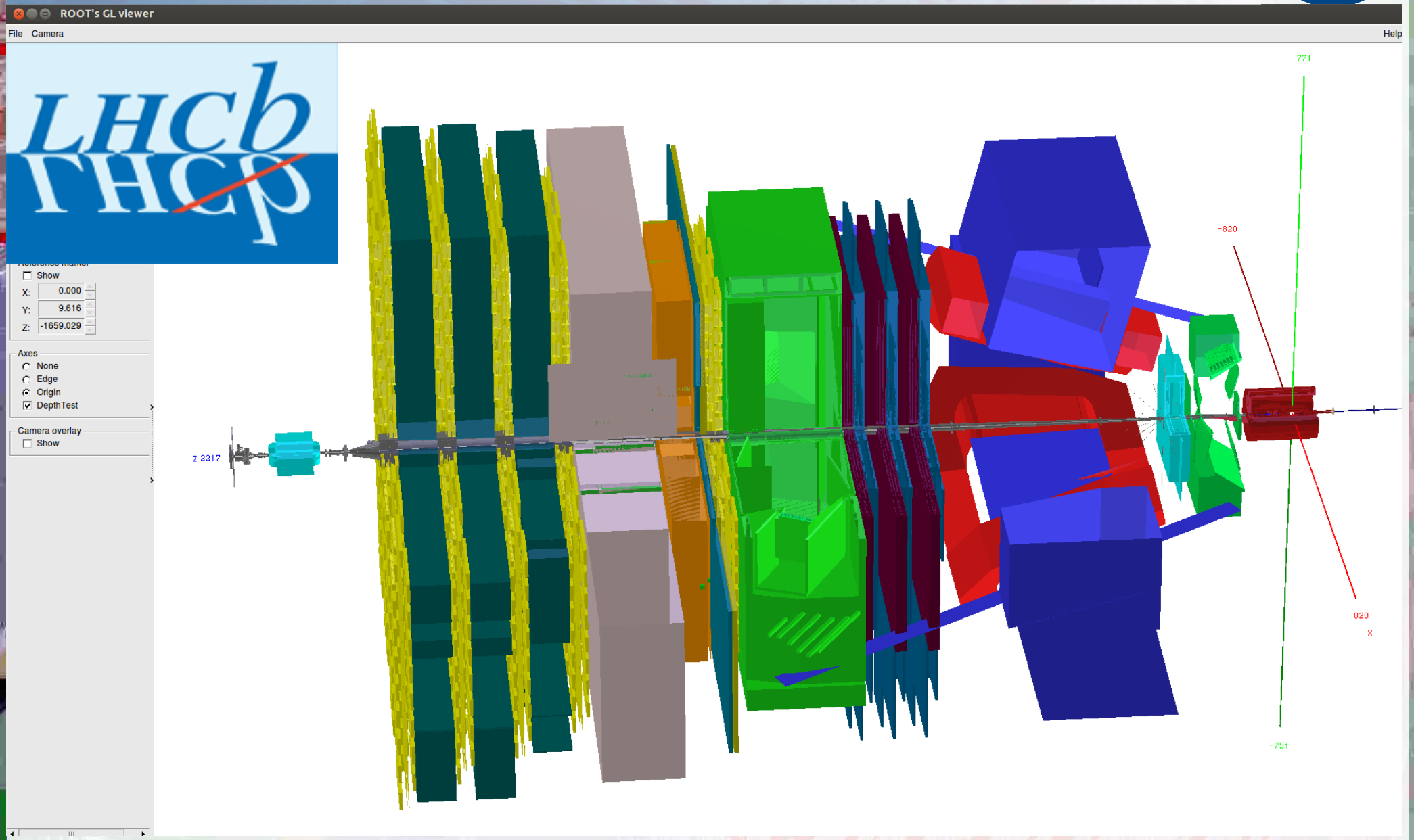
- **Use IOV dependent projection for event processing**
  - **This is our new "detector element"**
  - **Keeps reference to the not changing properties**
- **Dress with facade to provide required functionality(ies)**

PR-Plot: Barrel Tracker System



PR Plot: LHCb Detector of Run I / II

DD4hep



PR Plot: FCC Design Study

DD4hep

FCC
hh ee he