ATLAS Detector Data Model

Vakho Tsulaia
University of Pittsburgh

ATLAS – South Caucasus
Software/Computing Workshop & Tutorial
Tbilisi, 2010-10-25

Outline

- Introduction
- GeoModel toolkit for ATLAS Detector Description
  - Visual inspection and clash detection
- Geometry versions
- Usage of time-dependent detector conditions
  - Example: applying alignment corrections on top of 'frozen' geometry
- Usage of Detector Description (DD) in different contexts
  - G4 Simulation: Geo2G4 translator
  - Reconstruction: Tracking Geometry

Requirements for DD

- Very complex detector
  - Several sub-detector communities, need common environment for describing their unique geometries
  - Total of ~5 Million placed volumes in ATLAS full geometry description (GeoModel)
- Consistent geometry description for all interested clients
  - Reconstruction, Simulation
- Geometry versions
  - Many-to-Many correspondence between software releases and geometry versions
- Possibility to apply alignment corrections on top of the 'frozen' geometry
- Compact in memory and quick in initialization

GeoModel toolkit

- **GeoModel** – library of geometrical primitives that can be used for describing complex detector geometries
- GeoModel is a toolkit for ATLAS Detector Description software development since 2003-2004
- **Two layers** of geometry description
  - 'Raw' (material) geometry: physical volumes with transformations forming a geometry tree
  - Readout geometry synchronized to the raw geometry layer
- Single database as source for 'primary numbers'
  - Geometry database for building 'frozen' geometries
- Conditions database for applying alignment corrections

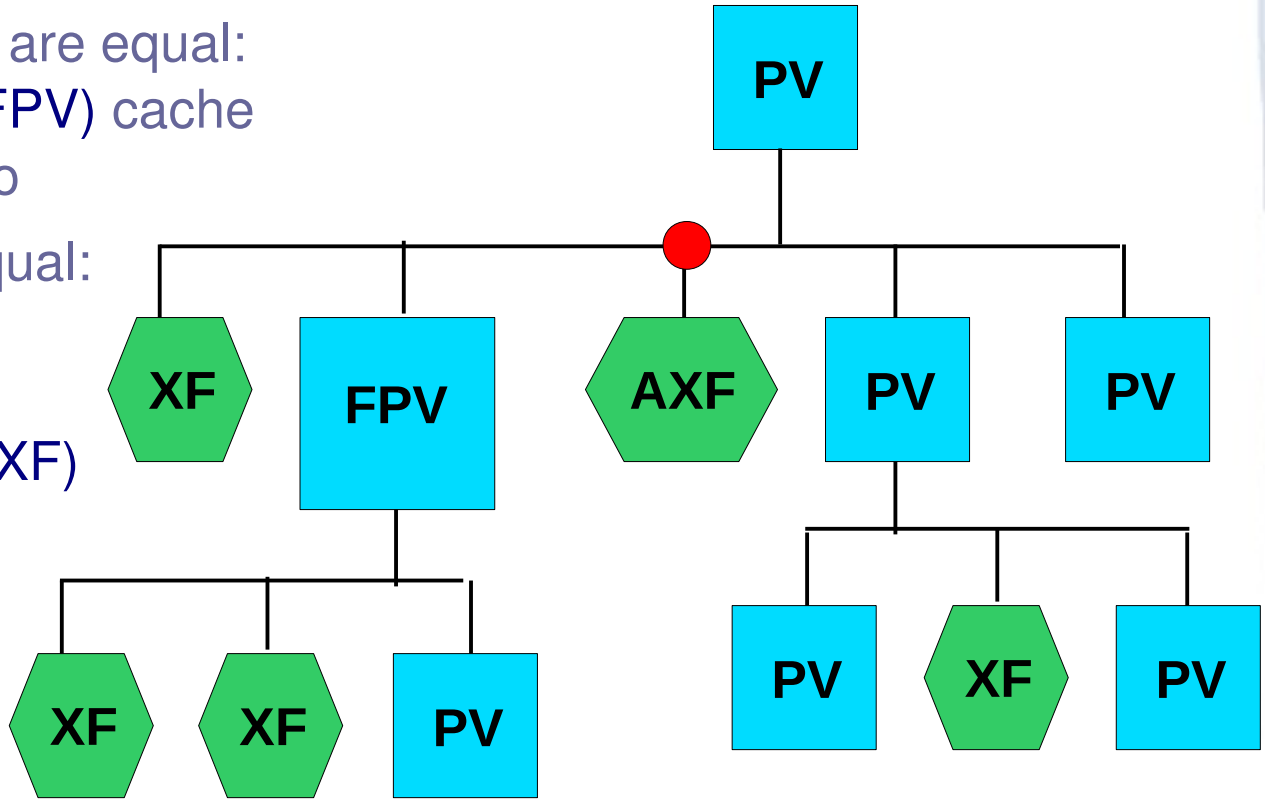
GeoModel: kernel library

- **Lightweight library**
  - Memory optimized geometrical primitives
  - Classes for building volume parameterizations
  - Volumes: Logical, Physical, Full Physical
  - Transformations: Standard, Alignable, Tiny
  - Base classes for readout geometry access
  - Identifiers: Names, Copy Numbers
  - Geometry tree navigation instruments: Actions
- **External dependencies:**
  - CLHEP
- **Documentation:**

http://atlas.web.cern.ch/Atlas/GROUPS/DATABASE/detector_description/Geometry%20Kernel%20Classes.doc

GeoModel: design principles

- The atomic unit of raw geometry description is the physical volume (PV)
- Its substructure: logical volume and transformation in parent's coordinate system
- Substructure of logical volume is Shape and Material
- Touch a transform (XF) and the subtree is moved
- Not all physical volumes are equal: Full Physical Volumes (FPV) cache global transformation info
- Not all transforms are equal: possibility to apply delta transformation to Alignable Transforms (AXF)

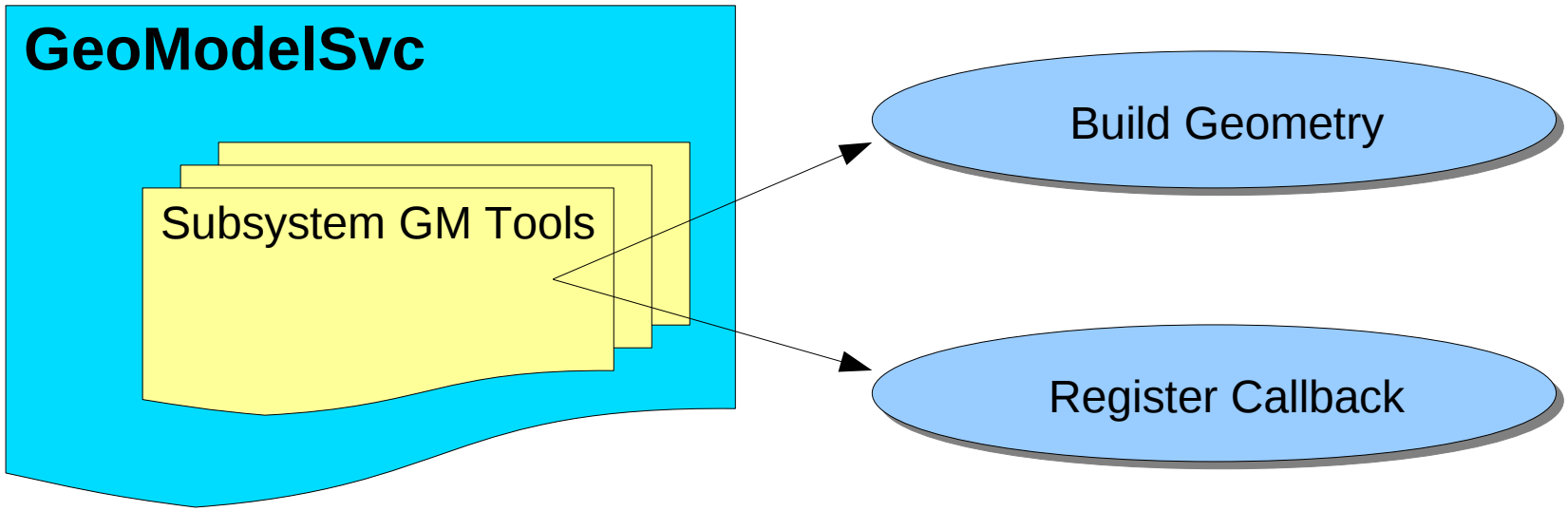


GeoModel: design principles

- **Readout geometry** is synchronized to raw geometry and is described with the use of **Detector Elements**
  - Detector Elements have required association with a piece of material geometry (Full Physical Volume)
  - The Detector Element interface can be extended by any detector specific information
- **Detector Manager** objects play the main role in providing an interface to Detector Description clients
  - One Manager per subsystem
  - Accessible to clients via Transient Detector Store
  - Holds pointers to subsystem's raw geometry tree top(s)
  - Implements subsystem-specific interface to access readout geometry

GeoModel in Athena

- ATLAS GeoModel applications integrated into Athena
  - Usage of Athena Tools and Services
- **5 Core** and **~30 subsystem** specific packages building entire GeoModel description of ATLAS
- Subsystem geometries built by subsystem specific **Tools**
- **GeoModelSvc** – Athena service managing subsystem tools



GeoModel: Visualization

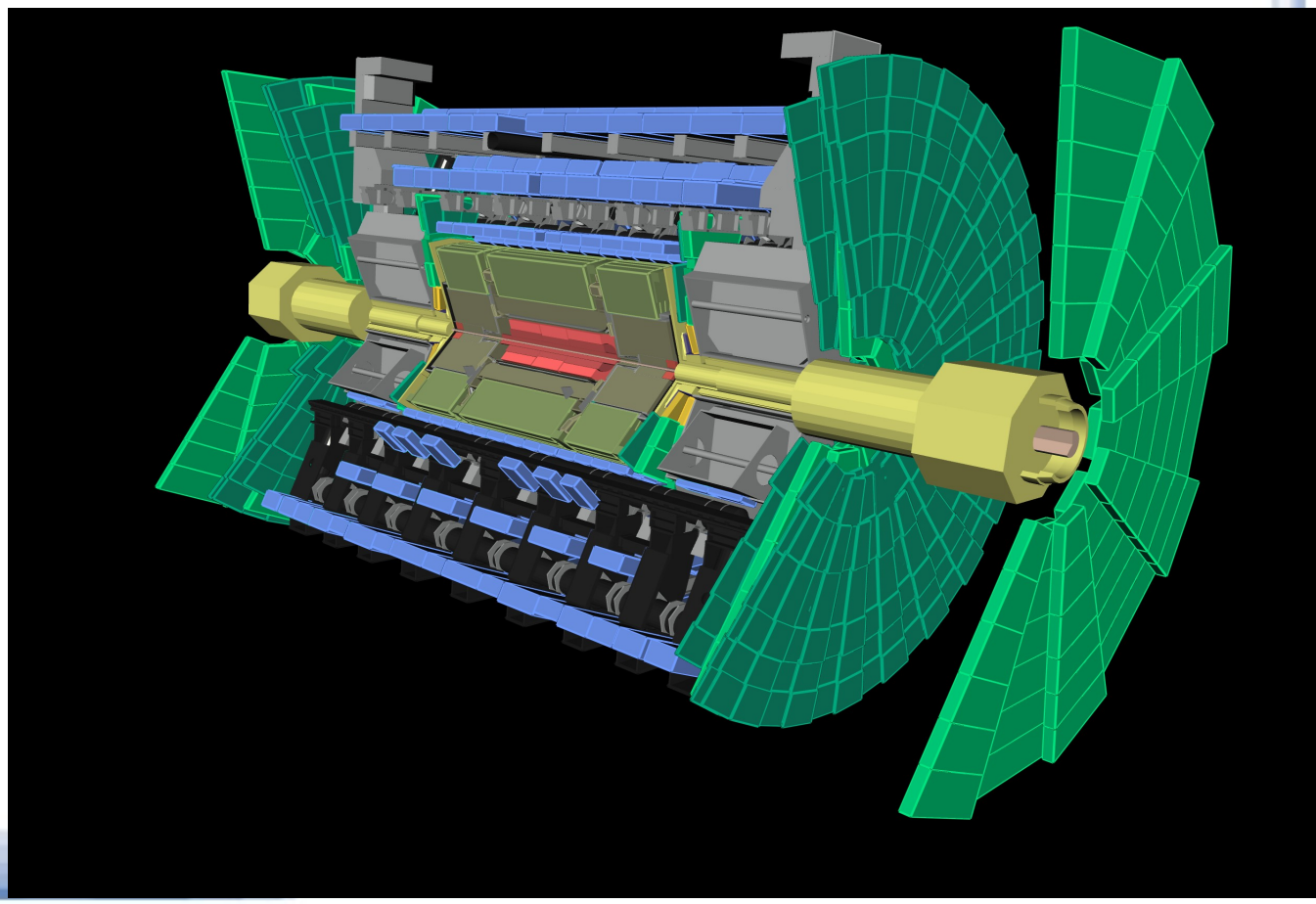
- **VP1** – ATLAS wide 3D Event Display

  http://cern.ch/atlas-vp1

- Displays detector geometry directly from GeoModel description
  - No extra conversions, no simplifications

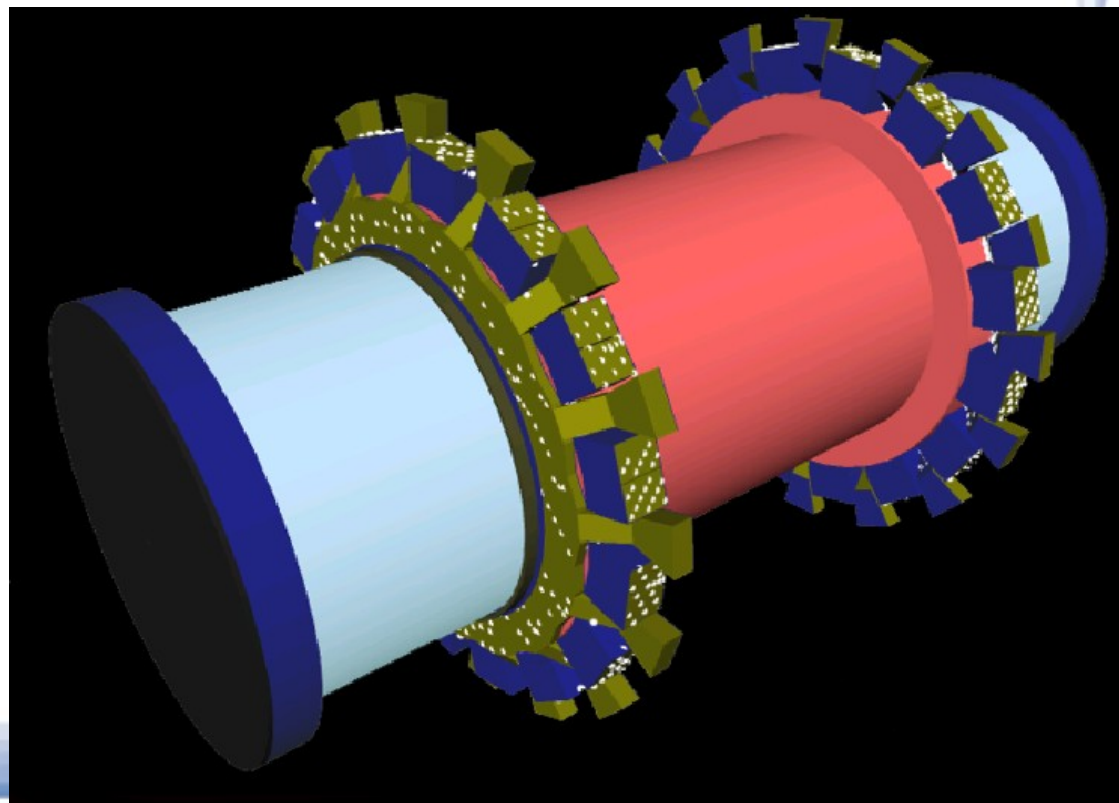
- Useful for debugging and understanding geometry description

- Many details about volumes:
  name,
  shape,
  transformation,
  material



GeoModel: detecting clashes

- Geometry description has to be **clash-free** ('clash' = volume overlap)
  - Especially important for simulation with Geant4
- **No native clash detector** available in GeoModel
- Use of **Geant4 geometry tester**
  - Build geometry in GeoModel
  - Translate it to G4 and run tester there
  - Analyze results and fix clashes
- Possibility to read geometry tester results in VP1 and co-display problematic points with detector geometry

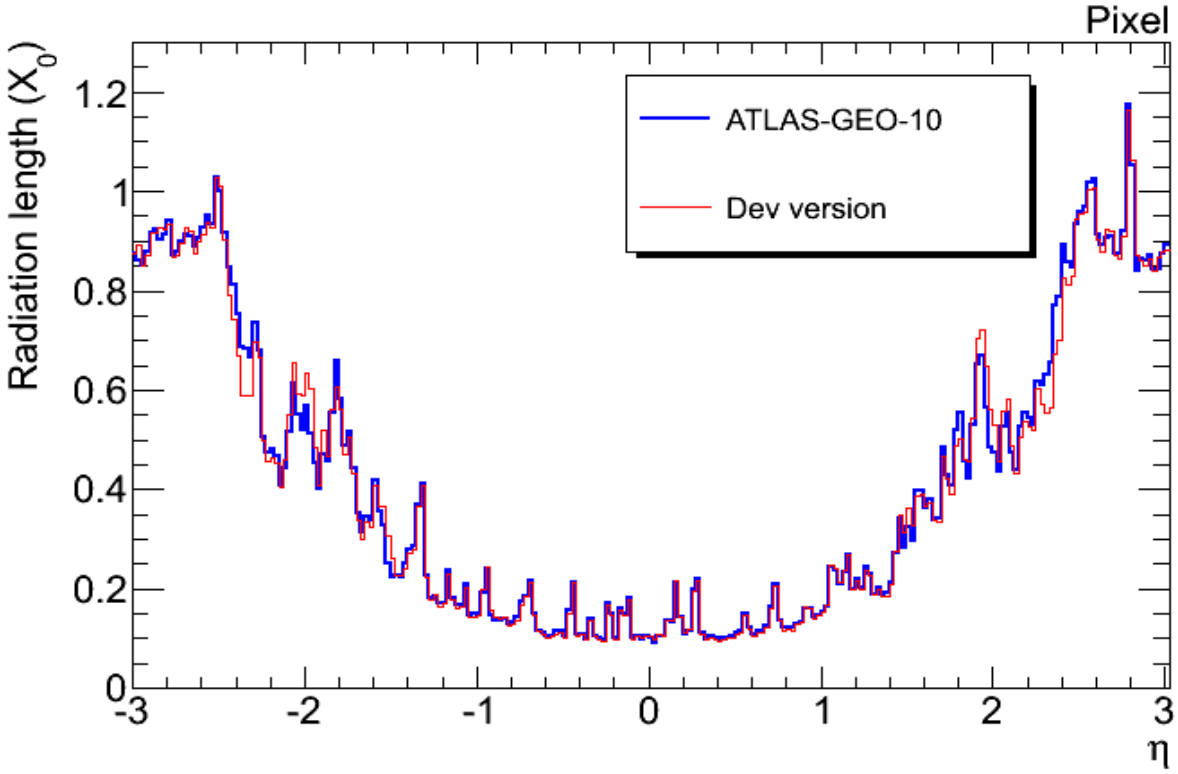


Geometry versions

- Possibility to build **various geometry configurations** with the **same software release**
- ATLAS geometry versioning system based on **versions of the primary numbers** in Geometry Database
- Many different configurations for various use cases
  - Realistic description on the entire detector geometry (ATLAS-GEO-XX layouts)
  - Material distortions on top of existing configurations
    - For studies by physics groups
  - Various Test Beam configurations (still used...)
  - Various prototypes for future SLHC configurations (IBL, MiniFCAL)
  - Various legacy configurations
    - DC2, CSC, etc
    - About to be declared obsolete

Geometry versions

- Geometry description constantly improving
  - Adding missing pieces
  - Improving existing material distributions
- Improvements and fixes go through thorough preparation and validation procedure before new geometry configuration gets frozer



Conditions

- Two sources of non-event data for Detector Description:
  - Geometry DB: Contains values that are constant over simulation cycle or data-taking period
  - Conditions DB: Contains values that change with a cycle, or get better determined between reconstruction phases
- Conditions DB is a large database containing information about detector status, calibrations, data-taking conditions, alignment etc...
- Data in the Conditions DB maps to transient C++ objects
  - Logically organized into tree of Folders (leaf) and Foldersets (branch)
  - Identified by IOV (start-stop)
  - And optionally by Tag

Conditions DB and Athena

- Athena uses **COOL** for accessing data in the Conditions DB
  - COOL part of the LCG Persistency Framework
- Special Athena service **IOVDbSvc** provides **interface** between **conditions data objects** and Athena Transient Detector Store (**TDS**)
  - Takes care of low level interactions with COOL
- IOVDbSvc ensures that **correct conditions objects are loaded** into TDS for the event being analyzed
- The interested clients
  - Either simply retrieve conditions objects from TDS
  - Or register callbacks to get notified when the object of interest changes in TDS
- Often direct access to conditions object is hidden behind interface of a specialized Athena Tool

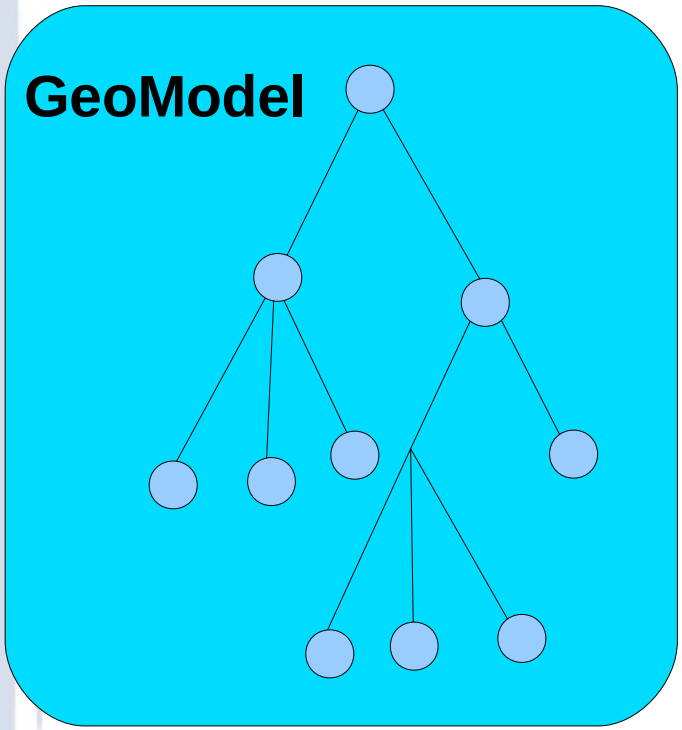
Alignment corrections

- GeoModel comes with an **intrinsic mechanism** of putting alignments on top of already constructed geometry
  - `setDelta()` method of the GeoAlignableTransform
- Subsystems store alignment information in the database in the form of (**Identifier,Transform**)
- Subsystem Tools or Detector Managers **register callbacks** on alignment objects (containers)
- When the callback is triggered, all necessary AlignableTransforms get updated in the geometry tree
- Next time client asks for the position of a Detector Element it gets a correctly updated result

GeoModel and Geant4 simulation

Geant4 needs detector geometry described with use of G4 native geometrical primitives

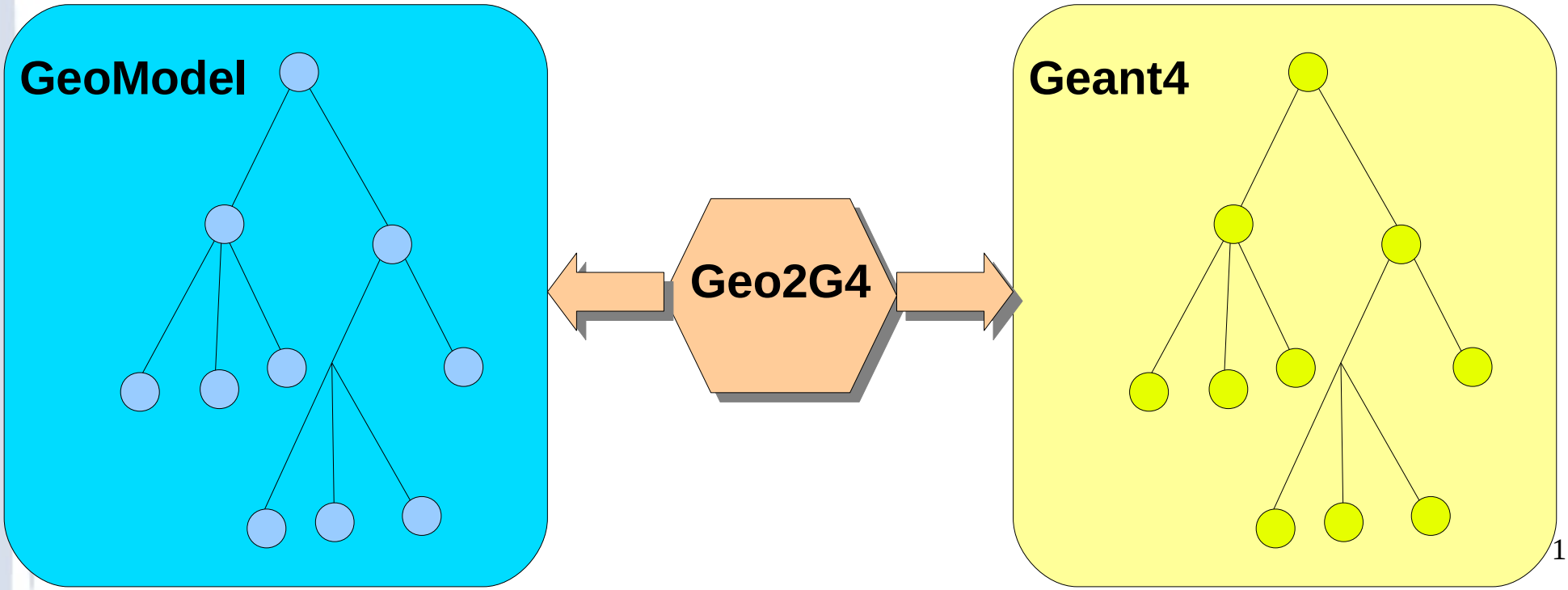
Step 1. Build detector geometry in GeoModel, apply all necessary alignment corrections



GeoModel and Geant4 simulation

Geant4 needs detector geometry described with use of G4 native geometrical primitives

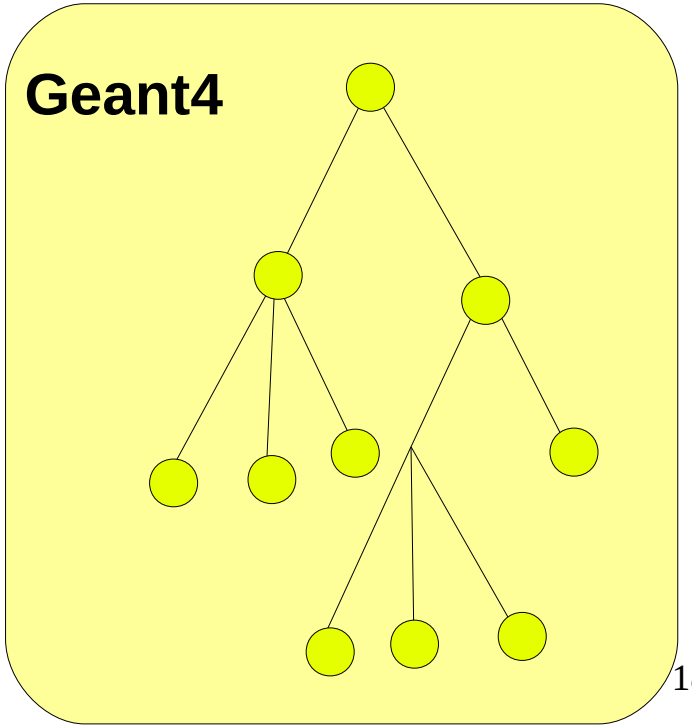
Step 2. Translate GeoModel geometry into Geant4 geometry with the use of Geo2G4 translator



GeoModel and Geant4 simulation

Geant4 has detector geometry described with use of G4 native geometrical primitives

Step 3. Release GeoModel description from memory



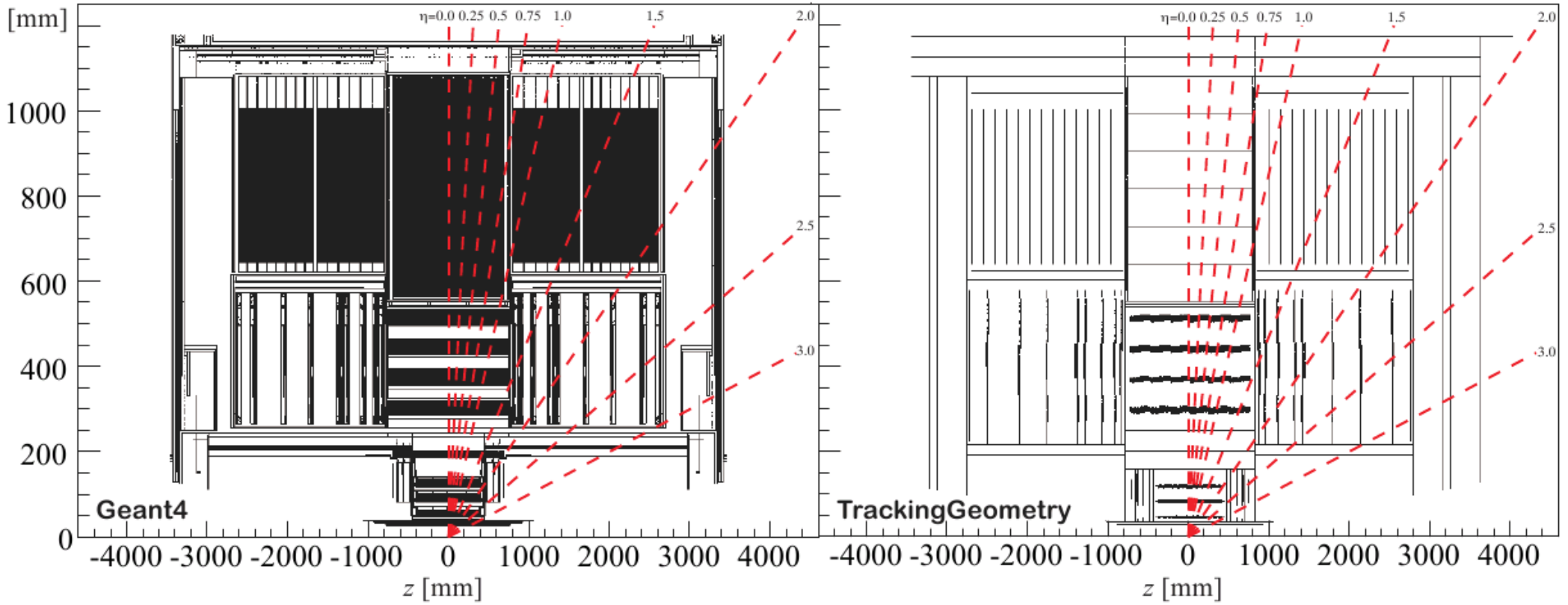
At this stage Geant4 is good to go

GeoModel and Reconstruction

- Reconstruction algorithms need to have access to the **readout geometry** layer
  - No need for direct access to the material GeoModel description
- Concept of '**minimal**' or '**reco**' GeoModel description
  - Build only those part of material GeoModel which are necessary for having functional readout description and for building Tracking Geometry (Next slide...)
  - This includes Full Physical Volumes referenced by Detector Elements
  - All AlignableTransforms need to be there too
- Performance benefits:
  - Faster initialization time: 40% improvement wrt full GeoModel
  - Smaller memory footprint: 25% improvement wrt full GeoModel

Tracking Geometry

- Very complex and detailed geometries not optimal for reconstruction
- **Tracking Geometry**: simplified geometry description parsed from GeoModel
  - InDet + Calo: cylinder and disk layers
  - Muon: Floating objects

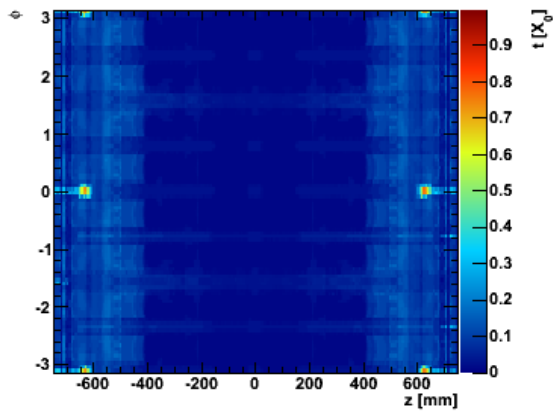


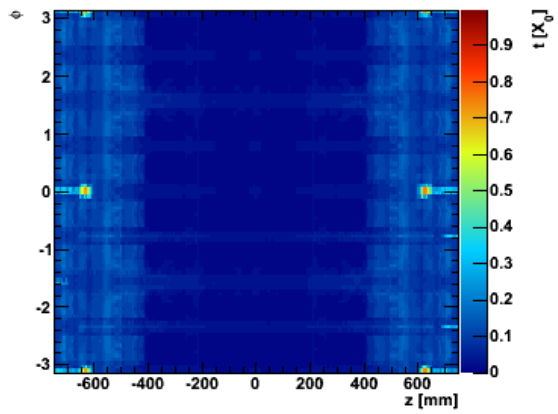
Inner Detector. Geant4 vs Tracking Geometry

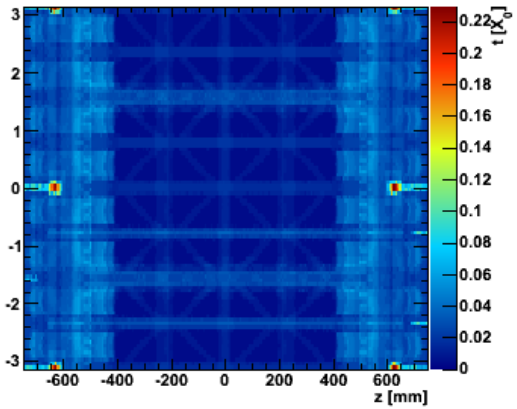
Tracking Geometry

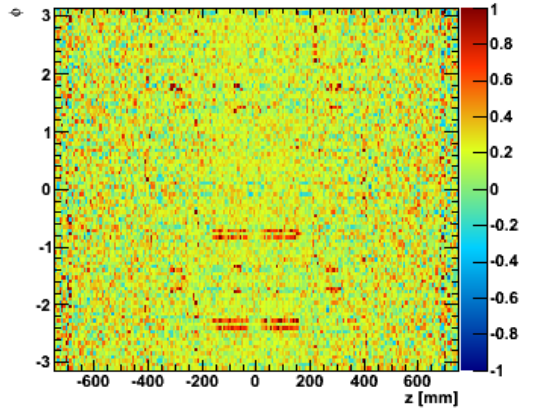
Material Integration

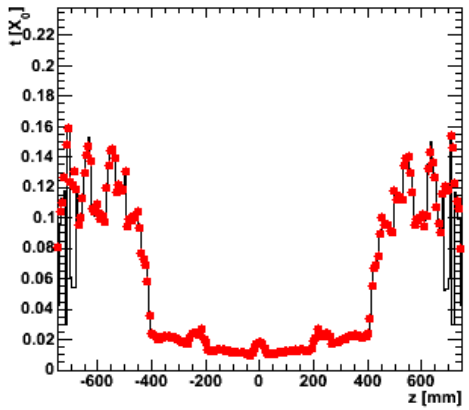
- Tracking Geometry provides fast material access for track reconstruction (to be included as stochastic noise in the track fit)
- Material maps for static layout (ID + Calo) from Geantinos projected onto the layers
- Material description for the Muon System by dense volumes (to be used with the STEP_Propagator)

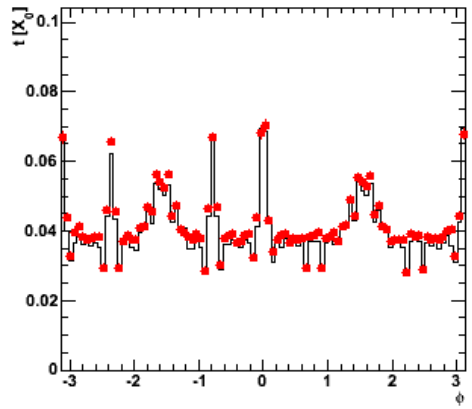












Comparison of G4/TrackGeo material for one reference layer

Conclusions

- ATLAS geometry described in GeoModel with a great level of details
- GeoModel provides single source of detector description for all interested clients
- The main activity of the ATLAS Core Detector Description has shifted from development to 'operations'
  - Integration of new subsystems: Forward Detectors, Components of the future SLHC configuration
  - Continuous creation of new geometry tags
  - Work on performance optimization issues
  - Code maintenance