Updating the software description of the ATLAS Detector

Marilena Bandieramonte[1], *Riccardo Maria* Bianchi[1,*], *Joseph* Boudreau[1], *Johannes* Junggeburth[2], *Evgueni* Tcherniaev[3], *Sarka* Todorova[4], *Vakho* Tsulaia[5], and *Rui* Xue[1]

[1]University of Pittsburgh (US)
[2]University of Massachusetts (US)
[3]CERN
[4]Charles University (CZ)
[5]Lawrence Berkeley National Laboratory (US)

Abstract. The software description of the ATLAS detector is based on the GeoModel toolkit, developed in-house for the ATLAS experiment but released and maintained as a separate package with few dependencies. A compact SQLite-based exchange format permits the sharing of geometrical information between applications, including visualization, clash detection, material inventory, database browsing, and lightweight full simulation. ATLAS simulation, reconstruction, and other elements of standard ATLAS offline workflows are now being adapted to ingest the geometry files, which are prepared using platform-independent modular geometry plugin code. This represents a major transformation of the ATLAS detector description software, impacting even the development procedures for which new roles have been invented. During these integration activities, both the GeoModel geometry kernel and the GeoModel toolkit have seen improvements, including volume calculation, material blending, helper classes for simpler memory management, and a richer collection of supported geometrical objects. This paper reports on these activities.

ATL-SOFT-PROC-2025-043
01 March 2025

1 Introduction

In the coming years, the Large Hadron Collider (LHC) at CERN will be updated, in the context of the "High-Luminosity LHC" (HL-LHC) project [1], to deliver higher luminosity. The upgraded accelerator will start operating in the LHC "Run4" data-taking phase in 2030–2033. In response to that, the ATLAS [2] detector will get major updates to cope with the larger number of collisions, increased number of generated particles, and the higher pileup. In particular, new detectors will be installed: new muon chambers [3], a new inner tracker (ITk) [4] that will replace the current Pixel [5], SCT [6], and TRT [7] trackers, and a High Granularity Timing Detector (HGTD) [8] (see Figure 1).

To ease the insertion and development of the new detectors and to ensure maintainability in the long run, the original ATLAS geometry package GeoModel [9] has been updated, and the overall Detector Description architecture of the ATLAS experiment has been recently

*e-mail: riccardo.maria.bianchi@cern.ch

reviewed, modernized, and optimized along the introduction of new tools[10]. This paper briefly presents the overall architecture of the new GeoModel toolkit, which is currently used by the FASER [11] and ATLAS experiments at CERN. Then, it highlights the latest tools that have been added to let developers easily develop, inspect, debug, visualize, and simulate detector geometries.

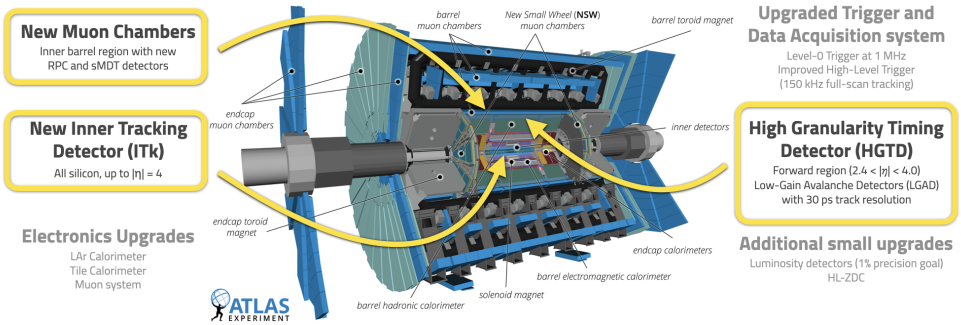

Figure 1. The figure shows the different components of the ATLAS experiment that will be upgraded for the upcoming "Run4"/HL-LHC. Those surrounded by a yellow border are new new particle detectors; the geometry of those are being developed with the new GeoModel toolkit.

2 The New ATLAS Detector Description

So far, the ATLAS experiment has used a Detector Description architecture [12] that is very well suited to the needs of the experiment in the Run 1, Run 2, and Run 3 LHC data-taking periods. However, the current architecture presents major limitations and flaws that prevent new detectors' easy insertion, development, and maintainability. In particular, as shown in Figure 2, in the current implementation, everything lives in the specific experiment's software framework, causing an inefficient and slow development cycle, with no flexibility in the choice of the tools to inspect, debug, and visualize the detector geometry. The current architecture is also based on multiple data sources, both in-framework and external, making it hard to track changes or tag and reproduce different versions of the geometry. The current geometry is also based on multiple ad-hoc interfaces for different detectors, making it hard to update or develop new ones, and challenging to maintain long-term (because of experts retiring or leaving the experiment). In addition to that, at the moment, the geometry "primary numbers", which are used to build the actual geometry volumes, are stored in a dedicated Oracle [13] database. Being a full-fledged database, it requires dedicated expertise to run, deploy, and maintain, making it hard for detector description developers to enter data independently. Also, not being open-source adds running costs to the experiment.

To overcome these limitations, the detector description for the HL-LHC/"Run 4" phase is based on a new architecture, whose schematic structure is shown in Figure 3. The GeoModel geometry kernel library and all tools needed to build the raw detector geometry have been moved out of the experiment's software framework. The Oracle DB will be decommissioned, and the geometry "primary numbers" will be defined in XML files and stored in a dedicated Git repository; the migration makes it easy for geometry developers to insert and update the "primary numbers" and all other related parameters, and it makes it easier for the detector description coordinators to track, tag, share, and reproduce different versions of the geometry. To store the complete geometry description, a new dedicated I/O layer has been developed

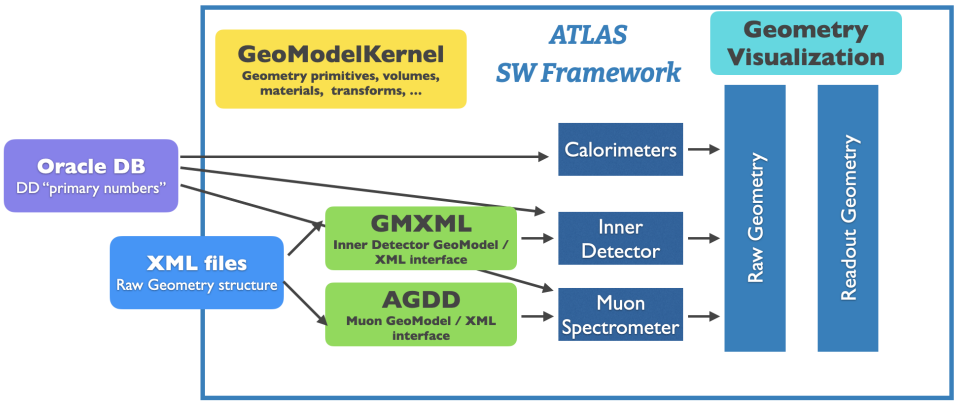

Figure 2. The figure shows the structure and the main components of the Detector Description architecture used by the ATLAS experiment in the "Run 1", "Run 2", and "Run 3" LHC data-taking periods.

to get a persistent copy of the GeoModel-based geometry into an SQLite [14] server-less database file. As a result, the raw geometry is now completely developed outside the framework, locally on the developer's machine, in a very fast and efficient development cycle, and there will be a single data input point for raw geometry into the experiment's framework.

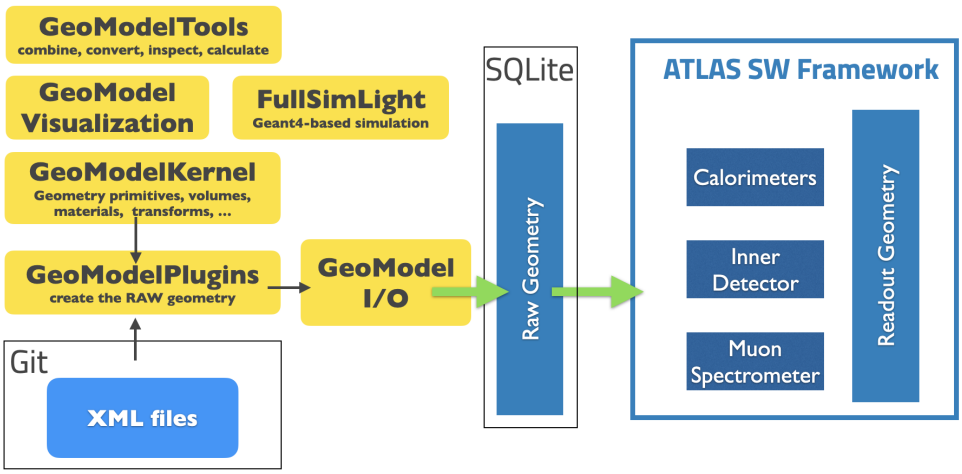

Figure 3. The ATLAS new, modular Detector Description architecture being developed for "Run 4"/HL-LHC, offering more flexibility, fastest development cycle, and improved code maintainability.

Several different tools have been developed besides the original geometry kernel library to ease the work of the geometry developers, as well as to design, build, store, handle, visualize, inspect, and debug detector geometries. The new packages compose a homogeneous, ever-growing, modular, pure C++ toolkit for HEP detector geometries, the GeoModel Toolkit [15]. Developers can choose which GeoModel modules to use, based on their needs, in a modular way and with minimal external dependencies. The original GeoModelKernel library, which

has been in use in the ATLAS experiment to describe all the detectors for more than 20 years, describes the detector as a tree of nodes: volumes, shapes, elements, materials, transforms, parametric placement, and other types of nodes. Most of the nodes can be shared to save memory, and the library features a very efficient mechanism for handling alignment, with multiple alignment constants kept in the cache. It also offers many memory-saving techniques, which help to get a very low memory footprint. As an example, the accurate geometry of ATLAS, which is the largest of the LHC experiments and composed of about 5 million nodes, only takes up about 80 Mb of memory. The I/O layer of the GeoModelIO package offers efficient I/O to dump and restore the whole tree of nodes, published volumes, alignable transforms, and any user-defined auxiliary data in SQLite files in an optimized way — As an example, the full ATLAS geometry is entirely contained in an SQLite file of about 20 Mb. The GeoModelTools package implements different tools to build, combine, convert, and translate geometries. At the same time, the GeoModelVisualization module offers the developer a Qt-based [16], interactive visualization tool that is optimized for geometry. Moreover, the FullSimLight[17] module provides a lightweight Geant4-based [18, 19] detector response simulation, as well as a graphical user interface ("FSL") to let geometry developers conveniently steer the simulation and a plugin mechanism to extend it [20]. The FullSimLight module also offers standalone programs to detect clashing volumes, create "geantino maps", and compute volume masses, among all the other tools and features.

The latest additions to the GeoModel toolkit are discussed in Section 3.

3 Recent Developments

3.1 New Memory Management

GeoModelKernel was first coded more than 20 years ago, when C++ only offered low-level memory management. At that time, a custom reference-counting mechanism was developed to create and delete GeoModel node instances into memory safely. However, over the years, the C++ programming language, which is used to develop the GeoModel toolkit, received many updates to ease memory management so that custom-made solutions are no longer needed. Therefore, recently, a major overhaul of the GeoModelKernel memory management has been performed, with the introduction of modern C++ smart pointers. That now ensures improved memory management as well as better maintainability of the GeoModel code base.

3.2 I/O Optimization, Improved Precision

The GeoModel toolkit offers an I/O mechanism to easily dump a persistent copy of a full GeoModel tree into a local SQLite database file and restore from it. The properties of all nodes in the tree are stored, as well as the relationships between them. When the development of the new I/O layer was started, the developers chose to treat all data in the database as mere strings, which were stored as TEXT data [14] in the database; that eased the development of both the interface and the database schema, letting different GeoModel node types share the same database table — notably, different Shape nodes, which were characterized of varying number of input parameters, were stored in a single table, and their parameters encoded in semi-colon-separated strings.

However, when dealing with real-life geometry data, the GeoModel developers realized that the numerical precision obtained by using strings as exchange format was not guaranteed when moving from one machine to another due to the different accuracy in the conversion between strings and real numbers on different architectures and operating systems; also, the needed precision was not assured in some cases.

To answer those issues, the database schema that is used in the GeoModel SQLite-based exchange format was recently heavily updated. The database now uses a different table for each type of GeoModel node, and, more importantly, all numerical data are now stored as REAL or INTEGER data in the database. The new schema not only improves portability between platforms guaranteeing the needed precision, it also offers room for more efficient parallelism, providing faster I/O and smaller size of the output database file.

3.3 New "Placed" Classes

Until now, in GeoModel, physical volumes were the only classes that could be placed in the 3D world because they owned their position; that made the development of new "placed classes" difficult.

To overcome that limitation, the internal hierarchy of the `GeoModelKernel` classes was updated to add intermediate classes (`GeoPlacement` and `GeoNodePositioning`) that hold the 3D placement and from which the "volume" classes now inherit, as shown in Figure 4.

That opened the possibility of having new additional types of "placed" classes. The first application of that is the introduction of "Virtual Geometry", which will be discussed in section 3.4.

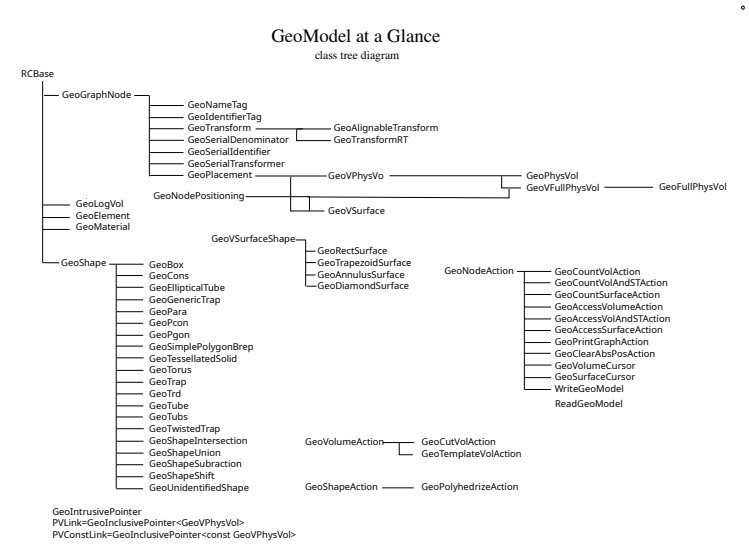

Figure 4. The new GeoModel class tree diagram showing the new `GeoPlacement` and `GeoNodePositioning` intermediate classes, letting developers create new "placed" classes.

3.4 Virtual Geometry

The reconstruction of tracks in ATLAS requires a detector geometry description to be used in track extrapolation processes as well as material effects integration during track finding and fitting. However, the complete detector description used in full detector simulation is too detailed, causing an unacceptable increase in CPU time consumption when used in track reconstruction. Because of that, a simplified version of the actual detector description is

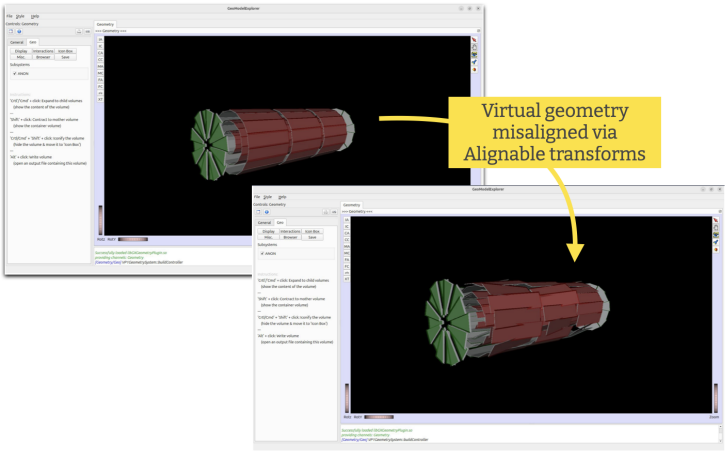

Figure 5. An example tracking detector built with the new Virtual Geometry surfaces and its components (mis)aligned by using `AlignableTransform` nodes (visualized in `GeoModelExplorer`).

preferable. That simplified "Tracking Geometry" is built of surfaces and is optimized for efficient navigation and fast extrapolation of tracks.

To describe and represent surfaces for tracking geometries with GeoModel, new "Virtual Geometry" classes were added to the GeoModelKernel library. Virtual geometry nodes are placed in the regular geometry tree, can be positioned and (mis)aligned alongside regular volumes, can be saved in the SQLite file and can be visualized as the regular geometry, as shown in Figure 5. Three surfaces have been added so far: (`GeoRectSurface`, `GeoTrapezoidSurface`, `GeoAnnulusSurface`); more surfaces will be added.

The most recent version of the ATLAS Tracking Geometry, which is being developed for the upcoming "Run 4" LHC data-taking period, is based on the detector-agnostic tracking software "ACTS" [21]. The introduction of the new GeoModel virtual surfaces helps to interface the ACTS- and GeoModel-based geometries and import ACTS geometries into the GeoModel tools to visualize, inspect, and debug them.

3.5 New Clash Detection

Detecting clashing and overlapping volumes is essential for accurate detector simulation, reconstruction, and alignment. The `gmclash` program is part of the GeoModel toolkit and is a standalone Geant4-based tool to detect clashes in GeoModel-based geometries. Clash detection is based on the Monte Carlo method, with random points generated on the volume's surface.

The tool was recently updated to let users set the number of random points and specify a given geometry subtree where clashes between volumes need to be detected. The detection algorithm was also updated to make it faster, and it is now based on the algorithm used in version 11.0 of the Geant4 simulation toolkit.

The `gmclash` tool generates as output a JSON-based file of clashing points, which can then be visualized in the standalone GeoModelExplorer (`gmex`) visualization tool.

3.6 New Shapes' Volume Calculation

Computing the volume of a geometry piece is necessary when dealing with material effects. New methods for volume calculation have been implemented for all GeoModel shapes, including boolean shapes. Different new utility tools were also added; they now let users get the bounding box of a shape and know if a given point is inside the shape. The new utility methods help determine the material used in the geometry volumes at a given point.

The database schema and interface of the GeoModel SQLite-based exchange format were updated, too, to store the volume of all shapes. This also lets users pre-compute the volumes of all shapes in a GeoModel tree and store them to later use them directly in computations. This also helps reduce the initialization time when volumes must be known at the start time of a detector simulation.

3.7 Support for Material Blending

New classes and tools were added to support "Material Blending". Different shapes can now be combined into a single simplified geometry to speed up the detector simulation, and all their materials can be blended into a single compound material. Based on those recent additions, the GeoModel developers are also designing new tools to exploit the material information in the GeoModel tree, compare it to engineering inputs, and produce simplified geometries for reconstruction and fast simulation; those will be the object of another paper.

3.8 Physics Validation

The core of the GeoModel toolkit, the `GeoModelKernel` library, has been used in ATLAS for more than 20 years as a base building block of the detector description. Therefore, the GeoModel core library has been *de facto* validated by 20 years of use in ATLAS and the many physics results published by the ATLAS Collaboration so far.

To ensure the correctness of the new detector description being developed for the aforementioned LHC "Run 4," the new architecture is currently being scrutinized through a rigorous "physics validation" process. In this process, the new Detector Description architecture is compared to the "old" architecture by examining numerous different physics variables.

Currently, the physics output of the Simulation step of the ATLAS data processing workflow [12] has been compared and validated. The Digitization and Reconstruction steps will follow.

In addition to accurate physics validation, the GeoModel toolkit's suite of automated tests has been expanded with new unit tests and regression tests. The documentation of the GeoModel toolkit [15] has been recently updated and expanded, too.

References

[1] CERN Yellow Reports: Monographs, "CERN Yellow Reports: Monographs, Vol. 10 (2020): High-Luminosity Large Hadron Collider (HL-LHC): Technical design report" (2020)

[2] The ATLAS Collaboration, "The ATLAS Experiment at the CERN Large Hadron Collider", Journal of Instrumentation **3**, S08003 (2008). 10.1088/1748-0221/3/08/S08003

[3] A. Policicchio, "The Phase-II upgrade of the ATLAS Muon Spectrometer", PoS **EPS-HEP2019**, 149 (2020). 10.22323/1.364.0149

[4] L. Gonella, "the ATLAS ITk detector system for the Phase-II LHC upgrade", NIM-A **1045**, 167597 (2023). https://doi.org/10.1016/j.nima.2022.167597

[5] B. Mandelli, "The Pixel Detector of the ATLAS Experiment for the Run 2 at the Large Hadron Collider", Nuclear and Particle Physics Proceedings **273–275**, 1166 (2016). 10.1016/j.nuclphysbps.2015.09.183

[6] The ATLAS Collaboration, "operation and performance of the ATLAS semiconductor tracker in LHC Run 2", Journal of Instrumentation **17**, P01013 (2022). 10.1088/1748-0221/17/01/p01013

[7] B. Mindur, "ATLAS Transition Radiation Tracker (TRT): Straw tubes for tracking and particle identification at the Large Hadron Collider", Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **845**, 257 (2017). 10.1016/j.nima.2016.04.026

[8] P. Fernandez Martinez, "Overview of the ATLAS High-Granularity Timing Detector: project status and results", PoS **EPS-HEP2023**, 525 (2024). 10.22323/1.449.0525

[9] J. Boudreau, V. Tsulaia, "The GeoModel toolkit for detector description", in *14th International Conference on Computing in High-Energy and Nuclear Physics* (2005), pp. 353–356, http://dx.doi.org/10.5170/CERN-2005-002.353

[10] M. Bandieramonte, R.M. Bianchi, J. Boudreau, A. Dell'Acqua, V. Tsulaia, "The new GeoModel suite, a lightweight detector description and visualization toolkit for HEP", Journal of Physics: Conference Series **2438**, 012051 (2023). 10.1088/1742-6596/2438/1/012051

[11] H. Abreu et al., "The FASER detector", JINST **19**, P05066 (2024). 10.1088/1748-0221/19/05/p05066

[12] ATLAS Collaboration (ATLAS), "ATLAS computing: Technical design report" (ATLAS-TRD-017) (2005), https://cds.cern.ch/record/837738

[13] The Oracle database, https://www.oracle.com/database/

[14] The SQLite database, https://www.sqlite.org/

[15] "GeoModel - A Detector Description Toolkit for HEP experiments", https://geomodel.web.cern.ch/

[16] The Qt framework, https://www.qt.io/product/framework

[17] M. Bandieramonte, R.M. Bianchi, J. Boudreau, "FullSimLight: ATLAS standalone Geant4 simulation", EPJ Web of Conferences **245**, 02029 (2020). 10.1051/epjconf/202024502029

[18] S. Agostinelli et al., "Geant4—a simulation toolkit", Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **506**, 250 (2003). 10.1016/s0168-9002(03)01368-8

[19] J. Allison et al., "Recent developments in Geant4", Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **835**, 186 (2016). https://doi.org/10.1016/j.nima.2016.06.125

[20] R. Khan, M. Bandieramonte, J. Boudreau, R.M. Bianchi, A. Dell'Acqua, D. Kleklots, V. Tsulaia, "recent developments in the fullsimlight simulation tool from atlas", EPJ Web of Conf. **295**, 03032 (2024). 10.1051/epjconf/202429503032

[21] C. Gumpert et al., "ACTS: from ATLAS software towards a common track reconstruction software", Journal of Physics: Conference Series **898**, 042011 (2017). 10.1088/1742-6596/898/4/042011