FullSimLight: ATLAS standalone Geant4 simulation

Marilena Bandieramonte[1,2,*], *Riccardo Maria* Bianchi[1,2], and *Joseph* Boudreau[1,2]

[1]CERN, EP Department, Meyrin 1211, Switzerland
[2]University of Pittsburgh, Dept. of Physics and Astronomy, Pittsburgh, PA 15260, USA

Abstract. HEP experiments simulate the detector response by accessing all needed data and services within their own software frameworks. However decoupling the simulation process from the experiment infrastructure can be useful for a number of tasks, amongst them the debugging of new features, or the validation of multi-threaded vs sequential simulation code and the optimization of algorithms for HPCs. The relevant features and data must be extracted from the framework to produce a standalone simulation application.
As an example, the simulation of the detector response of the ATLAS experiment at the LHC is based on the GEANT4 toolkit and is fully integrated in the experiment's framework "Athena". Recent developments opened the possibility of accessing a full persistent copy of the ATLAS geometry outside of the Athena framework. This is a prerequisite for running ATLAS GEANT4 simulation standalone. In this paper we present the status of development of FullSimLight, a lightweight simulation prototype that is being developed with the goal of running ATLAS standalone GEANT4 simulation with the actual ATLAS geometry.
The purpose of FullSimLight is to simplify studies of GEANT4 tracking and physics processes, including tests on novel architectures. We will also address the challenges related to the complexity of ATLAS's geometry implementation, which precludes make persistent a complete detector description in a way that can be automatically read by standalone GEANT4. This lightweight prototype is meant to ease debugging operations on the GEANT4 side and to allow early testing of new GEANT4 releases. It will also ease optimization studies and R&D activities related to HPC development: i.e. the possibility to offload partially/totally the simulation to GPUs/Accelerators without having to port the whole experimental infrastructure.

1 Introduction

High Energy Physics (HEP) experiments need to simulate their detector response with a high level of accuracy, to be able to correctly compare experimental data to simulated data from theoretical models. This goal is achieved executing a series of intermediate steps that range from the event generation to the Monte Carlo simulation of the interaction of particles with matter and of their transport through the detector, to the reconstruction of the physical quantities that represent the basic information to start the physics analysis. Each experiment has implemented its own custom solution and uses complex and large frameworks, through which

*e-mail: marilena.bandieramonte@cern.ch

it is possible to perform their entire data workflow. However, HEP experiments are not static and during their life-cycles many upgrade activities and R&D programs are in progress and the software frameworks need to cope with the new needs and new developments. For example, the upgraded detectors that the experiments at the Large Hadron Collider (LHC) foresee to commission in the near future, with new modules or cell-level shapes, finer segmentation, different materials and detection techniques, require additional developments of tools and features and imply new demands of physics coverage and accuracy. The experiments' upgrade activity encourages, for instance, the development, improvement and optimization of detector description tools that can make the modeling of complex detectors more efficient and computationally faster, providing new features, simplifying and speeding up significantly the development cycle. Each experiment carries out diverse R&D projects in order to successfully face all those challenges posed by near term and future research programs, everything within some constraints for example given by the available computing budget.

In this context, particle transport Monte Carlo simulation, that is implemented using the Geant4 toolkit [1–3], is considered as one of the main actors of the experiment simulation stack, being the main CPU consumer in most scenarios. Any development that can improve the performance of the Geant4 detector simulation, both in terms of speedup and accuracy, is of utmost importance. In this paper we introduce FullSimLight, a standalone lightweight ATLAS simulation prototype based on Geant4. It is conceived as a tool to streamline a number of tasks related to simulation and detector description, like the debug of new features and the early testing of new Geant4 releases or geometry clash detection. It could also be envisaged as a useful tool to ease the validation of multi-threaded vs sequential simulation code or the optimization of algorithms for HPC's environments. In Sec. 2 we will describe the complexity of Athena, the ATLAS software framework, in Sec. 3 we will describe the basic building blocks that are pre-requisites for running a standalone ATLAS simulation and in Sec. 4 we will present the status of development of FullSimLight, presenting some use cases. Finally in Sec. 5 we will draw some conclusions and present some future work.

2 Athena framework and ATLAS detector simulation

The Athena [4, 5] framework is the main software infrastructure used by the ATLAS [6] collaboration. It is based on the Gaudi framework [7] that supports a variety of applications through base classes and common functionalities, providing communication tools between different components. ATLAS collaboration uses the Athena framework to run the full software stack, that consists mainly of simulation, reconstruction and to some extent physics analysis, by iterating over an input dataset associated with single particles collisions, called events. The loop over each individual event is called the *Event Loop* and Athena controls the sequence in which different algorithms are executed inside the event loop, through the so called Athena *AlgSequence*. The full software chain can be summarized with the following steps:

- Step 1 - Event generation : Outgoing particles from LHC collisions are generated with specific energies and directions and stored in EVNT files
- Step 2 - Simulation : The generated particles are transported through the detector and their interaction with the detector components is simulated, producing the HITS files that represent the energy deposits of particles on the sensitive detectors cells
- Step 3 - Digitization: The ATLAS detector discrete response is calculated in forms of digits from the HITS file generated in the previous step, producing RDO files
- Step 4 - Reconstruction: Times and voltages representing the detector response are reconstructed into physics analysis objects and stored in AOD and ESD files

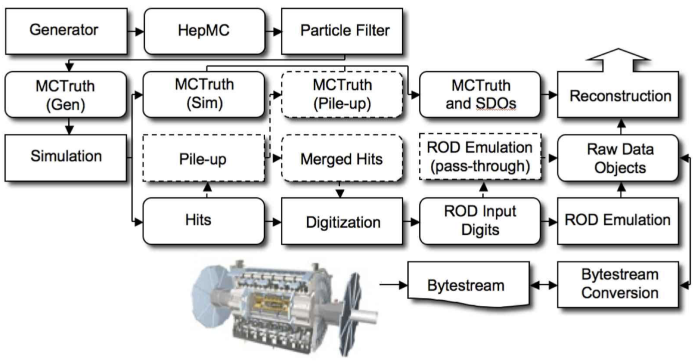


Figure 1: ATLAS Simulation infrastructure flow, from event generators (top left) to reconstruction (top right).

- Step 5 - Derivation and Physics Analysis: Only relevant data for a particular physics item is kept, producing DAOD files from the AODs. Physics analysis starts from those.

Athena-based programs are configured depending on the required data flow of the algorithms involved in the sequence. Jobs are configured through script files, so called *jobOptions*, that are generated by job transform wrapper scripts and use Python as front-end language. The ATLAS simulation software is being used for large-scale production of events on the LHC Computing Grid (but also on HPC and clouds). This simulation requires many components, from the generators that simulate particle collisions, through packages simulating the response of the various detectors and triggers. All of these components come together under the ATLAS simulation infrastructure [8], a very complex framework that manages many different elements. The flow of the ATLAS simulation framework is illustrated in Figure 1. The square-cornered boxes represents algorithms, while persistent data objects are placed in rounded boxes. The dashed lines represents the optional pile-up portion of the chain that is used only when events are overlaid. The generators are used to produce the outgoing particles from LHC collisions in HepMC formats and Monte Carlo truth is saved in addition to the hits that represent the energy deposition in the detector.

On one hand, the adoption of a common framework encourages the use of a standard approach, providing a skeleton of an application into which developers plug in their code and that embodies the underlying design and philosophy of the software. It is also useful to factor out common components and functionalities that can be re-used in different scenarios. However, having a unique software stack that controls and handles many different services and components, introduces extra-layers and extra-complexity that complicate some workflows and that could be avoided in some cases. The purpose of FullSimLight is to have a "light" standalone Geant4 simulation of ATLAS, light because no sensitive detector description is embedded into it. FullSimLight decouples the simulation process from the experiment infrastructure, moving it towards a standalone and platform-independent prototype. A tool like this can be very useful in terms of speeding-up some tasks and can also ease development cycles providing an agile tool that can be run easily over different platforms and independently from the main framework.

3 Towards a standalone ATLAS full simulation

In order to have a standalone simulation, relevant tools and data must be extracted from the main Athena framework. Some basic building blocks are required, the first of which is to have a standalone access to the ATLAS detector geometry. The historical detector description tool used by the ATLAS experiment is *GeoModel* [9]. Thanks to a recent development, *GeoModel* has been extracted from Athena, becoming a standalone package [10, 11] that is linked as an external project into Athena. With the standalone *GeoModel*, there is now also the possibility to make persistent any version/tag of the ATLAS geometry in a SQLite [12] database file and building it back from the same file using *GeoModelIO* [13, 14]. The second tool that is necessary to build a Geant4 simulation of the ATLAS detector, is a converter from *GeoModel* geometry to Geant4 geometry, the *GeoModel2G4* converter. This needed to be extracted from the Athena code too and made available in a standalone fashion. Third, we need a standalone access to the ATLAS magnetic field, if we want our tool to be useful for tracking studies. Having these basic building blocks one could write a lightweight Geant4 simulation, that builds the ATLAS detector, implements its magnetic field and runs in a standalone and platform-independent way. A possibility could be to go further with the I/O handling, i.e. events handling, conditions and alignment handling or HITS and sensitive detectors implementation, but this would go out of the scope of our project which is not meant to be a replacement of the simulation, but rather a tool to simplify R&D studies and testing activities.

3.1 Standalone *GeoModel* and Geometry Persistency

GeoModel is the detector description toolkit used in ATLAS: it describes standard shapes and nodes, which are then customized with parameters taken from an "online" Geometry DB (Oracle-based). Historically there wasn't the possibility to access a persistent copy of the ATLAS geometry, since the geometry was built on demand at run-time and kept only in memory. The initial motivation for having access to a standalone, persistent copy of the ATLAS geometry was the development of a light version of *VP1* [15], the ATLAS 3D event display tool integrated in Athena that can access all services and databases and can visualize all ATLAS data (ESD, AOD, EVGEN, HITS,...). This light version, called *VP1Light* [13, 16], is an event display visualization tool, that has been developed to run outside of the Athena framework on different platforms such as MacOS or Ubuntu. For that purpose the ATLAS geometry should be accessible outside of the Athena framework in a persistent format. This is what lead to the extraction of *GeoModel* from Athena and the development of routines to write-out and read-in a *GeoModel* geometry with *GeoModelIO*; a full toolkit suite based on *GeoModel* has been developed, including visualization of detector geometry and input/output tools. A persistent version of the full *GeoModel* tree as described in Athena, after the application of the parameters taken from the Geometry DB, is accessible in a standalone way and all the *GeoModel* libraries can be built outside Athena, easing the development of new code and its use in standalone applications.

3.2 *GeoModel2G4* converter and complexity of the ATLAS geometry

As said before, *GeoModel2G4* converter [17] is needed to convert the ATLAS geometry, that is read back from the persistent copy and built in *GeoModel* geometry format, into Geant4 geometry. For this purpose two main ATHENA packages are needed: *GeoMaterial2G4* and *Geo2G4*, which are respectively handling the material conversion and the shapes conversion.

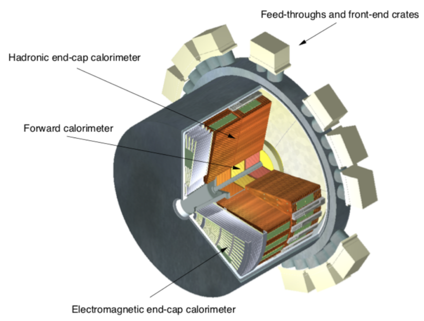

Feed-throughs and front-end crates
Hadronic end-cap calorimeter
Forward calorimeter
Electromagnetic end-cap calorimeter

Figure 2: Cut-away view of an end-cap cryostat showing the positions of the three end-cap calorimeters. The outer radius of the cylindrical cryostat vessel is 2.25 m and the length of the cryostat is 3.17 m.

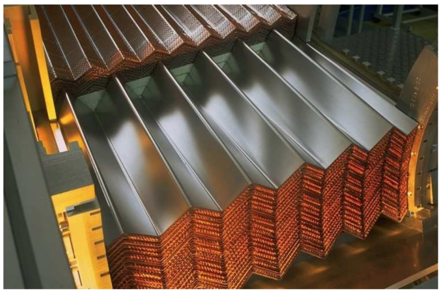


Figure 3: Picture of an electromagnetic end-cap module showing the accordion structure of the ATLAS EM calorimeters.

The extraction of the major part of these packages imply the extraction of a very specific ATLAS sub-detector with a custom, complex geometry: the ElectroMagnetic End-Cap (EMEC) special shape.

The EMEC [18] is a lead/liquid-argon sampling calorimeter with interleaved accordion-shaped absorbers and electrodes. Figure 2 shows a cut-away view of the end-cap cryostat with three calorimeters. During the experiment design phase, the geometry has been chosen of accordion type, shown in Figure3, for several reasons: i.e. it allows very good hermeticity, it increases the absorber rigidity which improves their relative positioning and hence contributes to the response uniformity, and it minimizes inductance in the signal paths. Due to the very complex geometric form of the EMEC detector, which is not represented by any stardard shape or volume implemented both in *GeoModel* and in Geant4, it has been chosen to describe it with a custom solid implementation [19]. For this reason it is not possible to directly make this shape persistent into standard geometry formats that are widely used in the HEP community such as GDML [20], or dump it in a way that can be automatically read by standalone Geant4. For this purpose it has been indeed necessary to export the code that implements the EMEC special shape in *GeoModel* as well as the code that implements the EMEC custom solid in Geant4. This work has been completed, so now we have the possibility to make persistent the whole ATLAS geometry, containing also the EMEC sub-detector, in SQLite files that can then be read-in in FullSimLight to build back the whole geometry and convert it into Geant4.

4 FullSimLight: status and features

FullSimLight [21] is a standalone Geant4 based simulation of the full ATLAS detector, built on top of *GeoModelCore* [11], *GeoModelIO*, *GeoModel2G4*, and Geant4 projects. It can run both in sequential and multi-threaded mode (the latter requires Geant4 to be compiled with multi-thread support). Figure 4 shows the different elements that compose this application.

The detector construction supports the following formats:

- SQLite geometry files: dump of ATLAS geometry, portions of it or other geometries
- GDML geometry files: dump of ATLAS geometry, portions of it or other geometries

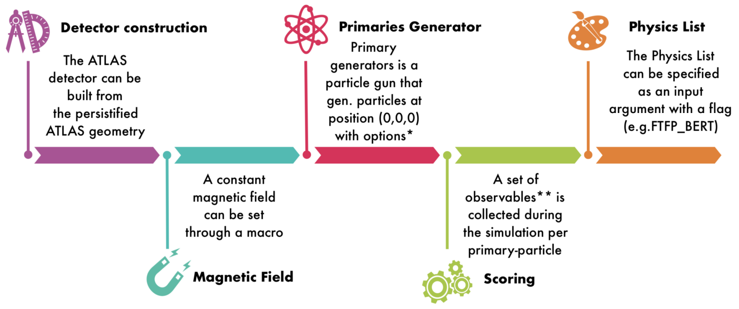

Detector construction
The ATLAS detector can be built from the persistified ATLAS geometry
Primaries Generator
Primary generators is a particle gun that gen. particles at position (0,0,0) with options*
Physics List
The Physics List can be specified as an input argument with a flag (e.g.FTFP_BERT)
A constant magnetic field can be set through a macro
Magnetic Field
A set of observables** is collected during the simulation per primary-particle
Scoring

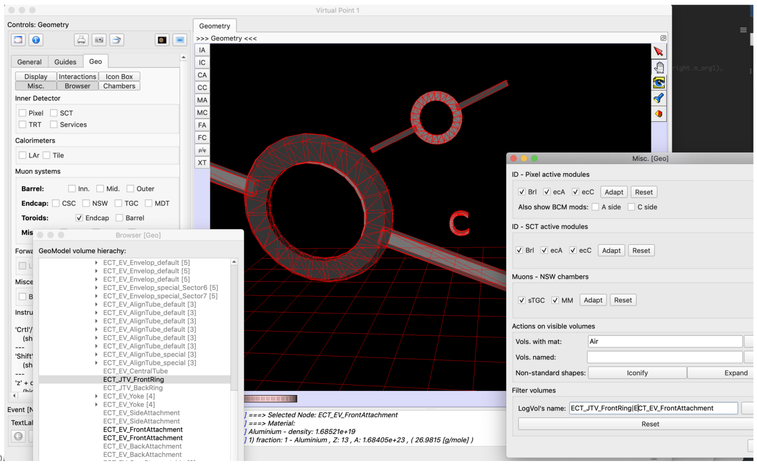
Figure 4: Sketch describing the main components of *FullSimLight* application at the present status.

- .so/.dylib plugins: self-contained packages describing a particular geometry

At the time of writing this paper a constant magnetic field is implemented in the application, and its value can be set through the default Geant4 input file. The integration of the ATLAS magnetic field map is in our near term working plan. *FullSimLight* uses a particle gun to generate particles at a specific position and with specific kinematic properties. Different physics lists can be specified and used for the simulation. A minimal set of "observables" is collected during the simulation per-primary particle type: mean energy deposit, mean charged and neutral step lengths, mean number of steps made by charged and neutral particles, mean number of secondary particles. The result is reported at the end of each event for each primary particle that was transported. At the end of the run some per-primary particle type simulation statistics of the above mentioned quantities are also reported. Sensitive detectors are not implemented, since *FullSimLight* is not conceived as a replacement of the entire Monte Carlo simulation, but rather as a tool that can be used to speed development up and testing activities, or R&D studies. We envisage the use of *FullSimLight* in many use cases such as studies of performance and optimization of Geant4 steppers for transport in magnetic field, easy production of *geantino maps* for ATLAS sub-detectors, studies about the implementation and use of Geant4 parallel geometries [22]: i.e. event biasing, scoring of radiation, and/or the creation of hits in detailed readout structures. Furthermore *FullSimLight* allows very easily to perform geometry overlap checks on each plugged-in geometry, taking into account the hierarchy of volumes which are organized in mother-daughters relationships inside the detector simulation. This is a very useful feature, that has already been used for debugging geometries and to detect clashing volumes or misplaced geometries. As an example, Figure 5, that has been produced with *VP1Light*, shows a clash of two volumes in a complex geometry, that has been detected running a geometry overlap check from within *FullSimLight*.

5 Conclusions

We have a first working prototype of *FullSimLight*: a standalone light MonteCarlo simulation of the ATLAS detector at the LHC. The relevant features have been extracted from the main experiment framework, Athena, to produce a first standalone simulation application. This lightweight prototype is meant to ease debugging operations on the Geant4 side and to allow early testing of new Geant4 releases, but it has shown already to be an useful tool for detector description geometry debug, in order to easily detect clashing volumes. In the near term



Inner Detector
☐ Pixel ☐ SCT
☐ TRT ☐ Services
Calorimeters
☐ LAr ☐ Tile
Muon systems
Barrel: ☐ Inn. ☐ Mid. ☐ Outer
Endcap: ☐ CSC ☐ NSW ☐ TGC ☐ MDT
Toroids: ✓ Endcap ☐ Barrel

Browser [Geo]
GeoModel volume hierachy:

```
ECT_EV_Envelop_default [5]
ECT_EV_Envelop_default [5]
ECT_EV_Envelop_default [5]
ECT_EV_Envelop_special_Sector6 [5]
ECT_EV_Envelop_special_Sector7 [5]
ECT_EV_AlignTube_default [3]
ECT_EV_AlignTube_default [3]
ECT_EV_AlignTube_default [3]
ECT_EV_AlignTube_default [3]
ECT_EV_AlignTube_default [3]
ECT_EV_AlignTube_default [3]
ECT_EV_AlignTube_special [3]
ECT_EV_AlignTube_special [3]
ECT_EV_CentralTube
ECT_JTV_FrontRing
ECT_JTV_BackRing
ECT_EV_Yoke [4]
ECT_EV_Yoke [4]
ECT_EV_SideAttachment
ECT_EV_SideAttachment
ECT_EV_FrontAttachment
ECT_EV_FrontAttachment
ECT_EV_BackAttachment
ECT_EV_BackAttachment
```

```
] ===> Selected Node: ECT_EV_FrontAttachment
] ===> Material:
] Aluminium - density: 1.68521e+19
] 1) fraction: 1 - Aluminium , Z: 13 , A: 1.68405e+23 , ( 26.9815 [g/mole] )
```

Misc. [Geo]
ID - Pixel active modules
✓ Brl ✓ ecA ✓ ecC Adapt Reset
Also show BCM mods: ☐ A side ☐ C side
ID - SCT active modules
✓ Brl ✓ ecA ✓ ecC Adapt Reset
Muons - NSW chambers
✓ sTGC ✓ MM Adapt Reset
Actions on visible volumes
Vols. with mat: Air
Vols. named:
Non-standard shapes: Iconify Expand
Filter volumes
LogVol's name: ECT_JTV_FrontRing|ECT_EV_FrontAttachment
Reset

Figure 5: Two clashing volumes in a complex Muon geometry, detected running a geometry overlap check with FullSimLight

FullSimLight will be used to simplify studies of GEANT4 tracking, allowing to test new steppers and GEANT4 features, to assess different performance gains using ATLAS detector and its magnetic field. In the long term we envisage the use of this prototype to produce *geantino maps* and ease optimization studies and R&D activities related to HPC development: i.e. the possibility to offload partially/totally the simulation to GPUs/Accelerators without having to port the whole experimental infrastructure.

References

[1] S. Agostinelli et al. (GEANT4), Nucl. Instrum. Meth. **A506**, 250 (2003)
[2] J. Allison et al., IEEE Trans. Nucl. Sci. **53**, 270 (2006)
[3] J. Allison et al., Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **835**, 186 (2016)
[4] P. Calafiura, W. Lavrijsen, C. Leggett, M. Marino, D. Quarrie, *The Athena control framework in production, new developments and lessons learned*, in *Computing in high energy physics and nuclear physics. Proceedings, Conference, CHEP'04, Interlaken, Switzerland, September 27-October 1, 2004* (2005), `https://cds.cern.ch/record/865624`
[5] *The Athena Framework*, `https://atlassoftwaredocs.web.cern.ch/athena/athena-intro/`
[6] The ATLAS Collaboration, Journal of Instrumentation **3**, S08003 (2008)
[7] G. Barrand et al., Comput. Phys. Commun. **140**, 45 (2001)
[8] The ATLAS Collaboration, The European Physical Journal C **70**, 823 (2010)
[9] J. Boudreau, V. Tsulaia, *The GeoModel toolkit for detector description*, in *Computing in high energy physics and nuclear physics. Proceedings, Conference, CHEP'04, Interlaken, Switzerland, September 27-October 1, 2004* (2005), pp. 353–356, `https://cds.cern.ch/record/865601`
[10] R.M. Bianchi, J. Boudreau, I. Vukotic, Journal of Physics: Conference Series **898**, 072015 (2017)

[11] *GeoModelCore*, `https://gitlab.cern.ch/GeoModelDev/GeoModelCore`

[12] *SQLite Library*, `https://www.sqlite.org`

[13] S.A. Merkt et al., EPJ Web Conf. **214**, 02035 (2019)

[14] *GeoModelIO*, `https://gitlab.cern.ch/GeoModelDev/GeoModelIO`

[15] T. Kittelmann, V. Tsulaia, J. Boudreau, E. Moyse, Journal of Physics: Conference Series **219**, 032012 (2010)

[16] R.M. Bianchi et al., Journal of Physics: Conference Series **898**, 072014 (2017)

[17] *GeoModelG4*, `https://gitlab.cern.ch/GeoModelDev/GeoModelG4`

[18] The ATLAS Electromagnetic Liquid Argon Endcap Calorimeter Group, Journal of Instrumentation **3**, P06002 (2008)

[19] J.P. Archambault et al., Tech. Rep. ATL-LARG-PUB-2009-001-1. ATL-COM-LARG-2008-002. ATL-LARG-PUB-2009-001, CERN, Geneva (2008), `https://cds.cern.ch/record/1095009`

[20] R. Chytracek, J. McCormick, W. Pokorski, G. Santin, IEEE Trans. Nucl. Sci. **53**, 2892 (2006)

[21] *FullSimLight*, `https://gitlab.cern.ch/GeoModelDev/FullSimLight`

[22] J. Apostolakis et al., *Parallel geometries in Geant4: Foundation and recent enhancements*, in *2008 IEEE Nuclear Science Symposium Conference Record* (2008), pp. 883–886, ISSN 1082-3654