A General Geometry System

BV

[2014-05-29 Thu 17:21]

1 Overview

This document provides a survey of existing geometry systems evaluated in terms of an general, ideal geometry system. Here, "geometry" refers to the description of the material (and possibly field) makeup of a region of space for the purpose of simulating or reconstructing the trajectory of particles through that space. We restrict the survey systems that employ a Constructive Solid Geometry (CSG) model.

2 Components

A geometry system can be conceptually broken down into the following components:

schema The meta-description of the data types used to describe any particular geometry.

representations A description may be represented in various forms: persistent files, transient, in-memory collections of objects or visually as a 2D or 3D static or interactive rendering.

converters The processes by which one representation may be converted into another.

producers Mechanisms and tools to create or alter a description, usually through visual or transient representations and resulting in persistent ones.

validation Mechanisms and tools which assure some level of correctness of a description. Validation can be performed to assure the persistent format follows a schema, that assumptions of the CSG model are

satisfied (ie, no overlapped volumes) or by human inspection through visualization.

management A mechanism by which a description may undergo controlled and recorded change.

An ideal geometry system will provide coverage for these concepts in a cohesive manner. All existing geometry systems considered in this survey have at least one of these components (by construction). None cover all. Some amount of inter-system compatibility exists and this allows for an aggregation of systems to cover more of the components.

3 Existing Systems

This section contains a survey of existing geometry systems selected because they are of use by High Energy Physics (HEP) experiments and are known to the authors. Each following section lists the features of the packages providing the system in terms of the components listed above. In some cases liberty is taken in defining what packages are considered together to make up a system. Any opportunity for inter-system compatibility is included where it may be applicable.

3.1 GDML

The Geometry Description Markup Language (GDML) consists of an XML schema and C++ and Python implementations.

3.1.1 Schema

GDML provides a schema as a number of XML XSD files. Persistent XML representations may be validated parsers that support XSD. GDML does not provide any schema for transient or visual representations.

3.1.2 Representations

The GDML Python or C++ bindings packages provide support for reading and writing persistent representations that follow the GDML XSD schema. Somewhat counter to the project's claim, there is no support for application-independent transient representation unless one counts the generic XML SAX data structures. The Python and C++ implementation support populating Geant4 or ROOT objects from these structures. The GDML packages provide no native representations for visualization.

3.1.3 Converters

As stated above, the Python and C++ implementation libraries provide converters between XML SAX representations of GDML files and Geant4 and ROOT transient objects.

The 3.6 package provides converters that can read and write GDML files and sink or source descriptions in other systems. ROOT supports reading and writing an older version of GDML files (fixme: which version?). Geant4 can be compiled against the C++ GDML library to allow conversion of GDML file to Geant4 objects and dumping of these objects to GDML files.

3.1.4 Producers

No tools for directly producing descriptions in this system are provided. GDML XML can be written with any text editor or dumped through the available converters.

3.1.5 Validation

XML validation on parsing is done based on the XSD. There is no validation of CSG assumptions nor included visualization.

3.1.6 Management

No facilities for managing descriptions are included.

3.2 AGDD

The ATLAS Generic Detector Description (AGDD) consists of an XML schema and Java library. Included as part of the system is the GraXML package .

is a text based format conforming to an XML schema. Like GDML it allows exhaustive geometry description as well as language structures for defining constant and variable parameters, loops, arrays. Files can be aggregated via the standard XInclude mechanism. Beyond a physical geometry description, AGDD has support for visualization by allowing colors to be defined on volumes.

3.2.1 Schema

3.2.2 Representations

3.2.3 Converters

3.2.4 Producers

3.2.5 Validation

3.2.6 Management

3.3 DetDesc

- DetDescCnv
- GiGaCnv

The Detector Description (DetDesc) is a text based format conforming to an XML DTD. Like GDML and AGDD it allows exhaustive geometry description as well as language structures for defining constant and variable parameters, loops and vectors. It allows aggregation through the standard XML external entity mechanism as well as a unique reference mechanism analogous to and named identically to HTML's `href`. It has been extended to allow for injection of XML at run-time through environment variables.

3.3.1 Schema

3.3.2 Representations

3.3.3 Converters

3.3.4 Producers

3.3.5 Validation

3.3.6 Management

3.4 ROOT

3.4.1 Schema

3.4.2 Representations

3.4.3 Converters

3.4.4 Producers

3.4.5 Validation

3.4.6 Management

3.5 Geant4

This "system" includes GDML support and all the vis methods.

- Geant4 Geometry XML Converter

  `http://root.cern.ch/root/vmc/XML.html`

3.5.1 Schema

3.5.2 Representations

3.5.3 Converters

3.5.4 Producers

3.5.5 Validation

3.5.6 Management

3.6 VGM

Converters, converters and more converters.

3.6.1 Schema

3.6.2 Representations

3.6.3 Converters

3.6.4 Producers

3.6.5 Validation

3.6.6 Management

4 Summary

This section to include

- summary of the landscape, discussion of pros/cons/tradeoffs of different packages
- A summary "matrix" of packages vs aspects with some kind of yes/no or numerical quality value in each cell

5 Ignore this section

This section holds some temporary notes.

5.1 Links

Maxim's agdd/gdml comparison `http://www.star.bnl.gov/public/comp/simu/GDML/gdml_vs_agdd.html`

Maxim's STAR simulation pages `http://www.star.bnl.gov/public/comp/simu/newsite/`

5.2 Conceptual schema

All existing schema considered here share a large overlap in schema at the conceptual level. They all use a "solid model" as apposed to a "surface model" for their description. This model contains these main data objects:

shape a.k.a. "solid" is a region of space in a specific share and size with a conventionally chosen coordinate system.

material the collection of bulk physical properties (density, isotopes, optical properties, sometimes visualization properties).

boundary a property related to the boundary of two materials, but not associated with either (for example, to model reflection in a non-Lorentz way)

volume the aggregation of a shape, material and possibly boundaries as well as other volumes. A sub-volume is placed w.r.t. a volume by specifying a translation and rotation. Typically any sub-volume shapes must be completely contained in the volume shape and no sub-volume shapes may overlap. This sub-volume placements produce a volume hierarchy.

touchable a path through the volume hierarchy in order to indicate some semantic importance to a particular region of space.