GraXML*

J.Hřivnáč†, LAL, Orsay, France

Abstract

GraXML is the framework for manipulation and visualization of 3D geometrical objects in space. The full framework consists of the GraXML toolkit, libraries implementing Generic and Geometric Models and end-user interactive front-ends. GraXML Toolkit provides a foundation for operations on 3D objects (both detector elements and events). Each external source of 3D data is automatically translated into Generic Model which is then analyzed and translated into Geometric Model using GraXML modules. The construction of this Geometric Model is parametrised by several parameters (optimization level, quality level, ...) so that it can be used in applications with different requirements (graphical or not). Two visualization applications are provided in the GraXML framework: GraXML Interactive Display and GraXML Converter into various 3D geometry formats. Other applications can be easily developed.

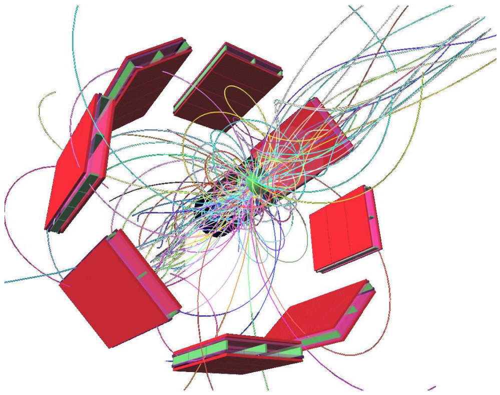

Figure 1: GraXML

GRAPHICS

Requirements on Event/Detector Display

Requirements of Event/Detector Display correspond to requirements on Geometric Modeler:

- Logical navigation in complex geometric structures,
- Geometrical navigation:
  - Where am I ?
  - Intersections (= picking, collision detection),
  - Materials (= colors),
  - Displacement and calibration (= interactivity).
- Optimized geometry (millions of 3D objects).

The only additional Display functionality is the manipulation of visual properties (transparency, shading,...).

3D Geometry Engine can be used as a Geometric Modeler foundation if visual overhead can be removed - Java3D is such Geometry Engine.

Graphical Domains

Graphics for HEP consists of two domains with different requirements and strategies for visualization and optimization:

Event Display contains Event related objects (Tracks, Hits,...). They are usually represented by simple (Hits,...) or unique (Tracks,...) geometrical objects. Special (2D) projections are most suitable to display them.

Detector Display contains Detector elements (Chambers, Detectors,...). They are usually represented by replicated complex geometrical objects. Real 3D graphics with shared representations is required for their display.

GraXML covers that split by modularization of object models (both Generic and Geometric) and creating specific models for each domain.

Geometric data are first processed in a specific way according to their nature and optimal models are created from them. Those models are then visualized by the graphical program.

GraXML can display all data (Events as well as Detectors), its main mission is 3D visualization.

GRAXML

GraXML Design is described on the Figure 2.

Capabilities

GraXML (in graphical mode) offers full functionality for browsing and manipulating 3D data. All the standard User Requirements functionality for 3D display is available:

- Standard 3D visual operations, like Rotation, Translation, Zooming, Scaling, Sheering and Skewing. Those operations can be anisotropic, if it is applicable. All those operations can be applied either to the whole Scene (Global Operations) or to the selected (picked) objects (Local Operations). Local operation allows, for example, to move one object away from others. Operations are available via Drag mouse action, using keyboard or through the menu. Go to operation is

* http://home.cern.ch/hrivnac/Activities/Packages/GraXML

† Julius.Hrivnac@cern.ch

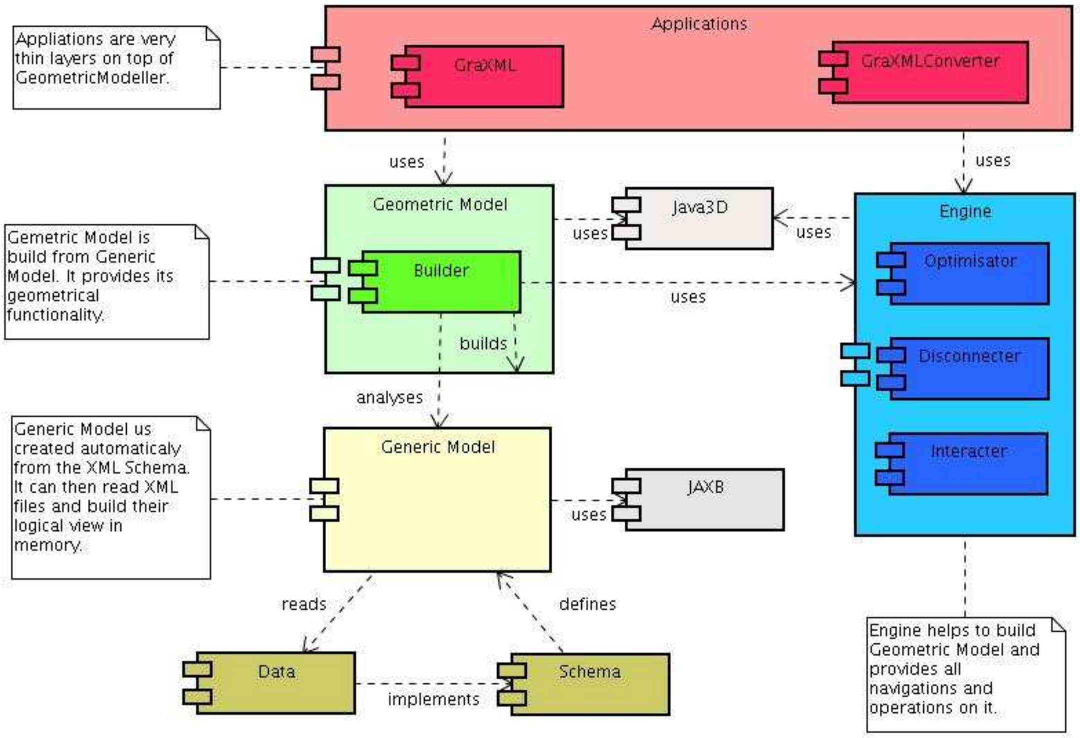

Figure 2: GraXML Component Diagram.

also available, it moves observer close to the selected object (or away from it).

- Changing of visual properties of an object. The color of an object, its transparency properties and its polygon status (solid - wireframe - vertexframe) can be changed interactively. Operation can be applied either to the selected (picked) objects or the the whole SceneGraph.
- Both Parallel and Perspective projections.
- Cutting the 3D Scene from front and from back.
- Cutting the 3D Scene by movable plain.
- Selective switching of the objects on and off from the hierarchical Tree Navigator. The switching works also in the optimized mode, the result may be, however, surprising due to hidden relations.
- Snapshot saving into the jpg file.
- Context sensitive help and Context sensitive information of the Tree Navigator elements.
- Optimized or fully Interactive navigation. The Optimized mode uses much smaller memory footprint and is therefore useful for the larger 3D Scenes (more than tens of thousands of simple objects). Interactive features are limited in this mode.
- Quality level. Higher quality levels offer features like better antialiasing or smoother objects, but requires significantly more CPU and memory.
- Output to VRML [1] and X3D [2]. VRML output can be also used as an input into a POV-Ray [3] Raytracing format file, which can be then used to create photographic-quality pictures (incl. reflections,...).
- Full Java scripting interface. Complete Java environment is available in a command-line scripting interface.

Foundations

GraXML is build on top of several foundation packages: **Java3D** [4] gives complete 3D geometry, with or without graphics. **HEP3D** [5] (GraXML contribution to FreeHEP [6]) contains Java3D primitive shapes (all Geant4 [7] simple elements, Tracks, outlining elements, colors, etc.). **Java3DBool** [8] allows boolean operations (union, intersection, difference) on 3D elements. **Colt** [9] performs efficient mathematics and primitive collections. **CyberVRML** [10] handles Java3D export to VRML. **BeanShell** [11] provides command line and scripting in Java. **JAXB** [12] interprets Generic Model in natural Java objects.

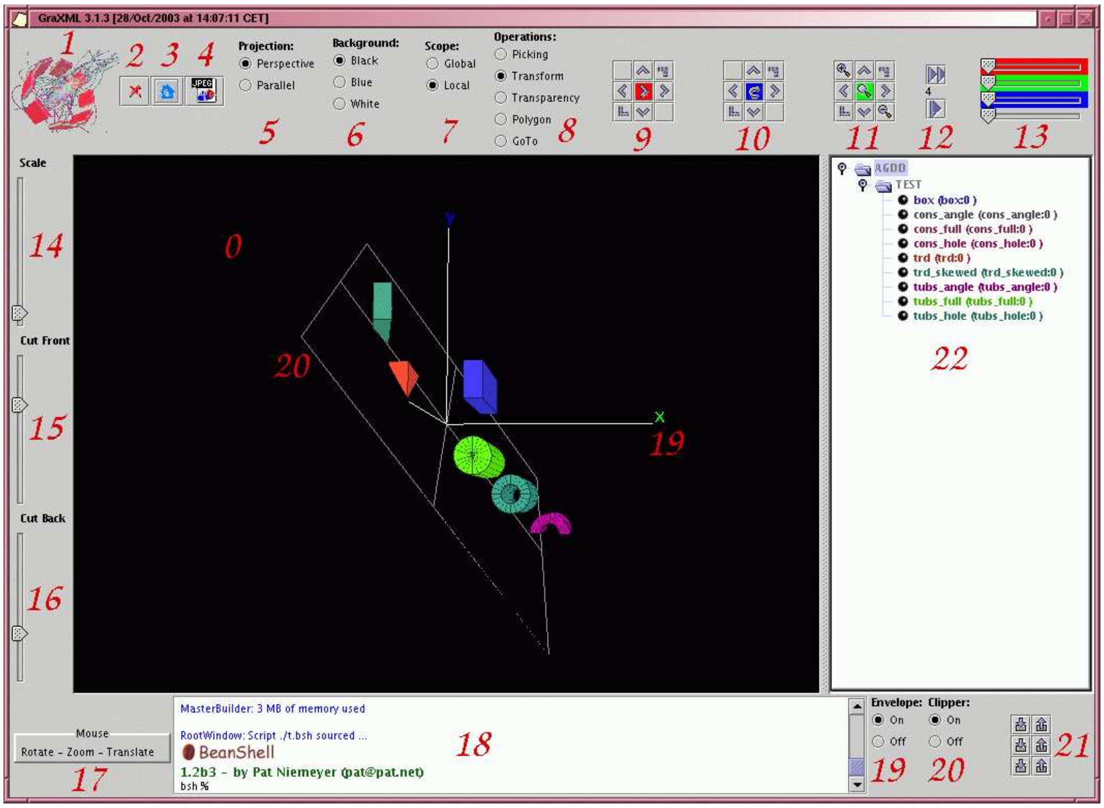

Figure 3: Example of running GraXML Display.

User Guide

GraXML Display GUI (see Figure 3) provides following functionality:

1. Shows About information as ToolTip.
2. Exits. Writes also Transformations selected in 21.
3. Goes Home. Resets the view into the standard one.
4. Saves Jpg picture.
5. Selects Perspective or Parallel Projection. Zooming doesn't have any effect in Parallel Projection, Scaling (14) doesn't have any effect in Perspective Projection.
6. Selects color of Background.
7. Selects Global or Local Operations (8). Global Operations are applied to the whole pictures, Local Operations are applied only to selected (Picked) objects. Some Global Operations don't have sense (Picking, GoTo). Local Operations are only partially available for Optimized SceneGraph.
8. Selects Operations connected with Picking. Operations connected with mouse buttons are described in the context sensitive help (17).
   - Picking: Just picks objects and reports their information in 18.
   - Transform: Performs standard transformation operations.
   - Transparency: Changes transparency status of objects.
   - Polygon: Changes polygon (Solid-Wireframe-Points) status of objects.
   - GoTo: Changes view so that selected objects is well visible.
9. Translates along any axes.
10. Rotates along any axes.
11. Scales along any axes or zooms in/out (magnifying glass buttons). Scaling deforms objects.
12. Defines value of one step performed by operations in 9-10-11.
13. Changes color or transparency status of selected objects.
14. Scales whole scene. It is much faster then zooming, but it doesn't have any effect in Perspective Projection.
15. Cuts the scene from front.
16. Cuts the scene from back.

17. Gives context sensitive help for actual actions defined for all buttons.

18. Exposes Java scripting interface and gives feedback (for example about picked objects). While full access to running Java environment is available, only non-graphical operations are documented and supported (mostly writing and reading files and setting of options).

19. Switches Envelope (Axis) on/off.

20. Switches interactive Clipper on/off.

21. Allows for selecting three predefined Transformations (views). Left button selects current Transformation, right button applies the selected Transformation. Selected Transformations are written into file when GraXML is ended via the Exit button (2). This file (which exists per-directory) is read back when GraXML is started next time.

22. Allows for interactive Showing/Hiding of elements or elements groups. In the case of an Optimized SceneGraph, the effect may be sometimes surprising for a user as elements are linked together (some are just reference to others) and are shown/hidden in groups. Object' description reflects its graphical status (bold - visible, normal partly visible, italic - invisible). ToolTip gives more details about object' properties.

SOURCES

Generic Model

XML Schema is used to create JAXB Generic object Model, which is then instantiated from XML files.

JAXB model is more natural (and faster) then DOM/SAX models as it maps all XML elements to classes with corresponding structure (in DOM, all XML elements are mapped into one class).

GDML [13] (Geometry Description Markup Language) files are supported at the level of version 1.1.0.

Geant4 geometry can be exported to GDML using VGM [14] (Virtual Geometry Modeler) or VMC [15] (Virtual Monte Carlo). Root geometry can be exported using VMC.

AGDD [16] (Atlas Generic Detector Description) files are supported at the level of version 7+, i.e. including mathematical formulas, all Geant4 simple solids and connection to SQL database of primitive numbers.

AGDD is frozen.

Jive (Atlantis [17] Event Display) files are supported at the prototype level.

Jive files Schema has been partially re-constructed and implemented.

HEPRep[18] files are not yet supported.

Geometric Model

Geometric Model is created from Generic Model following several options:

- Graphical or not (i.e. with or without graphical overhead).
- Level of optimization
- Level of quality.
- Level of interactivity.
- Choice of representations.

REFERENCES

[1] VRML - Virtual Reality Modeling Language (http://www.vrmlsite.com)

[2] X3D - XML 3D file format (http://www.web3d.org)

[3] PovRay - Persistence of Vision Raytracer (http://www.povray.org)

[4] Java 3D Engine (http://java.sun.com/products/java-media/3D, http://www.java3d.org, http://www.j3d.org, https://java3d.dev.java.net)

[5] HEP3D - FreeHEP 3D Graphics Foundation (http://home.cern.ch/hrivnac/Activities/Packages/HEP3D)

[6] FreeHEP - HEP Components and Tools for Java (http://java.freehep.org)

[7] Geant4 - Toolkit for the simulation of the passage of particles through matter (http://wwwasd.web.cern.ch/wwwasd/geant4/geant4.html)

[8] The Boolean Set Operations Project (http://www.geocities.com/danbalby)

[9] Colt - Libraries for High Performance Scientific and Technical Computing in Java (http://dsd.lbl.gov/ hoschek/colt)

[10] CyberVRML - Virtual Reality Modeling Language Development Library (http://cybergarage.org/vrml/cv97)

[11] BeanShell - Lightweight Scripting for Java (http://www.beanshell.org)

[12] JAXB - Java Architecture for XML Binding (http://java.sun.com/xml/jaxb)

[13] GDM - Geometry Description Markup Language (http://gdml.web.cern.ch/gdml)

[14] VGM - Virtual Geometry Model (http://ivana.home.cern.ch/ivana/VGM.html)

[15] VMC - Virtual Monte Carlo (http://root.cern.ch/root/vmc/VirtualMC.html)

[16] AGDD - Atlas generic Detector Description (http://www.wlap.org/atlas/computing/workshops/jan2001/agdd)

[17] Atlantis - Atlas Event Display (http://atlantis.web.cern.ch/atlantis)

[18] HEPRep - Generic Interface Definition for HEP Event Display Representables (http://heprep.freehep.org)