GEANT4 Geometry
Status & Plans

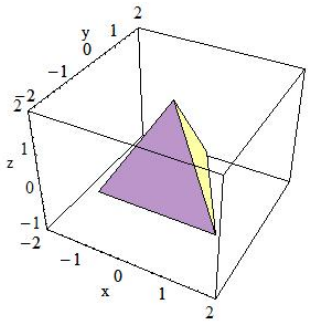
Witold Pokorski, CERN

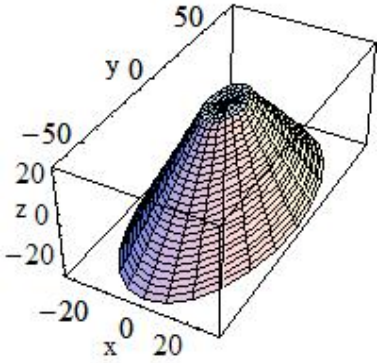
Outline

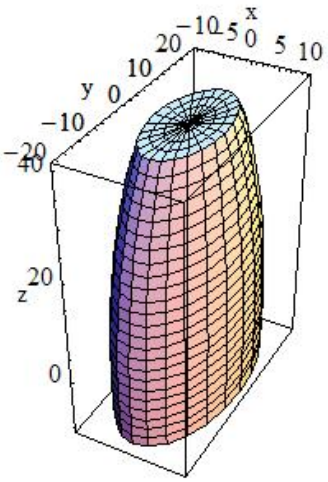
- Relevant developments in release 8 series
- Focus on
  - Detection of overlaps
  - Tessellated solids
- Short-term planned developments
- Status and new developments in GDML
  - Installation and dependencies
  - New features and future plans

Geometry modeler - 8.0

- Default constructor for direct object persistency (see slide 9) W.Pokorski
  - For volumes, solids, materials
- Generation of random points on surfaces D.Anninos, O.Link, V.Grichine
  - For all combination of solids, including Boolean operations
- Detection of overlaps at placement/construction of volumes G.Cosmo
  - Activated optionally in the constructor or on-demand afterwards
- New shapes: G.Guerrieri, D.Anninos, M.H.Mendenhall
  - Ellipsoid, elliptical cone, tetrahedra
- Nested parameterised volumes J.Apostolakis
  - Special construction for multi-dimensional parameterisations

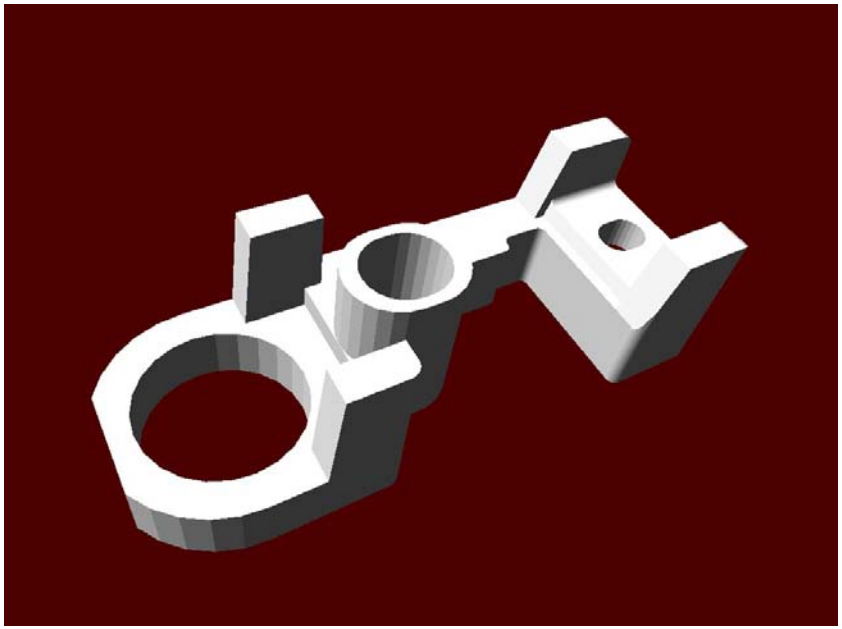






Geometry modeler - 8.1

- Extensions to `G4TransportationManager` G.Cosmo
  - To support future development for multiple navigators
- Extensions to G4AssemblyVolume I.Hrivnacova, G.Cosmo
  - To support assemblies of assemblies
  - To support reflected volumes
  - To allow for overlaps check at construction also for reflections
  - To allow for access to constituent volumes
- First implementation of tessellated solids
  - Volumes defined by triangular or quadrangular facets P.R.Truscott
  - Allowing import/export of shapes/assemblies with CAD systems
- Generalised implementation of store notifiers G.Cosmo
  - To be adopted in future also for materials



Detection of Overlaps

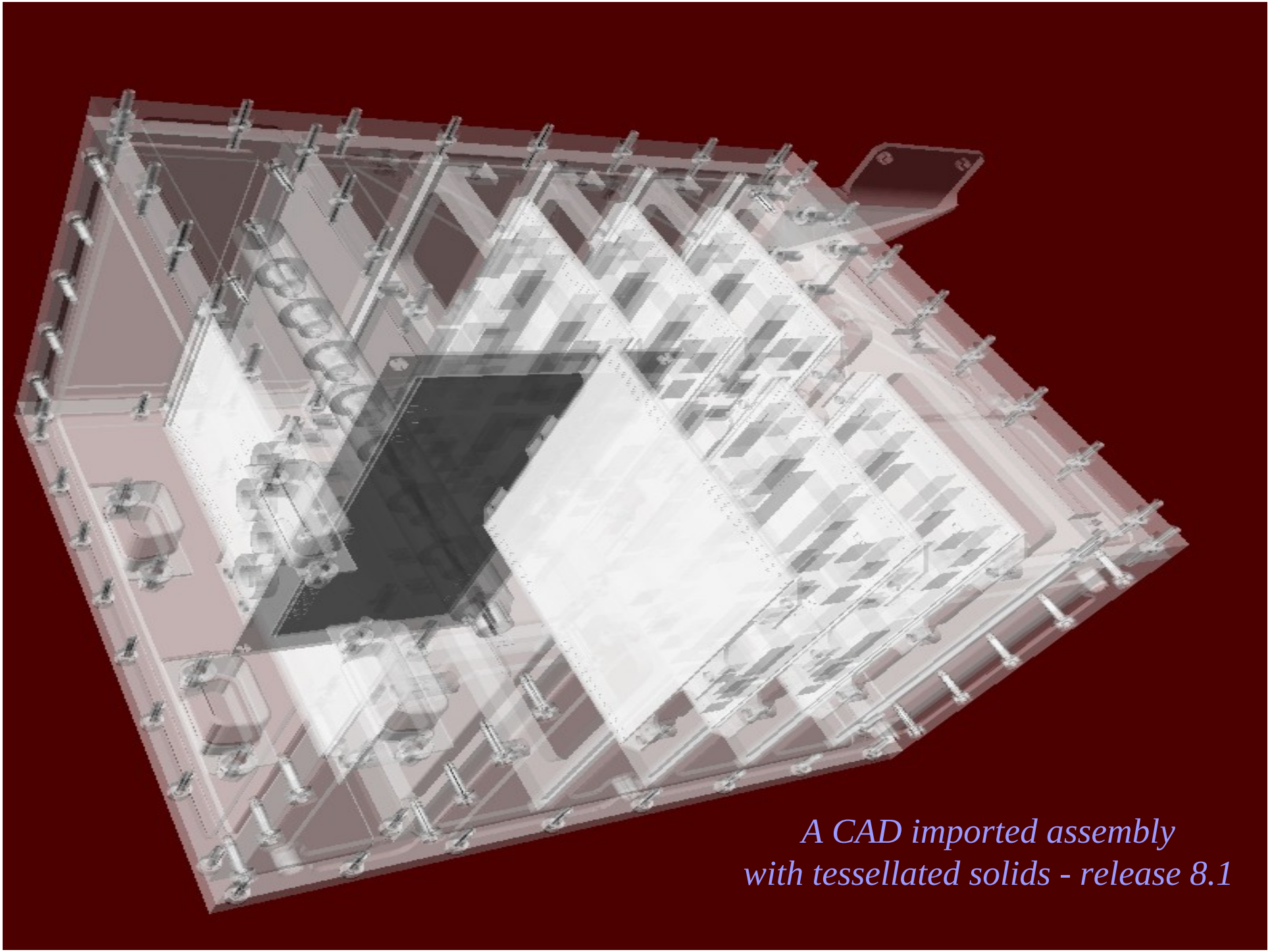
- Existing techniques on *constructed* geometry
  - Grids overlap (built-in UI commands)
    - Uses solids response and tracking
  - DAVID tool
    - Using polyhedron graphical representation
  - OLAP tool
    - Verifying tracking response through rays
- *All post-debug techniques*
  - Applicable to subsets for complex setups

Detection of Overlaps - 2

- Check overlaps at construction (since 8.0)
  - Applicable to placements and parameterised
  - Applicable for assemblies and reflections
  - Verifies placement of a single volume against the existing placed volumes
    - Activating a flag in the constructor
    - Or using explicit **`CheckOverlaps(int res)`** method
    - Generates 1000 points (**`res`**) on surface as default
- Allows for easy checks for misalignments

Tessellated solids

- **`G4TessellatedSolid`** (since 8.1)
  - Generic solid defined by a number of facets (**`G4VFacet`**)
    - Facets can be triangular (**`G4TriangularFacet`**) or quadrangular (**`G4QuadrangularFacet`**)
  - Constructs especially important for conversion of complex geometrical shapes imported from CAD systems
  - But can also be explicitly defined:
    - By providing the vertices of the facets in *anti-clock wise* order, in *absolute* or *relative* reference frame
  - GDML binding



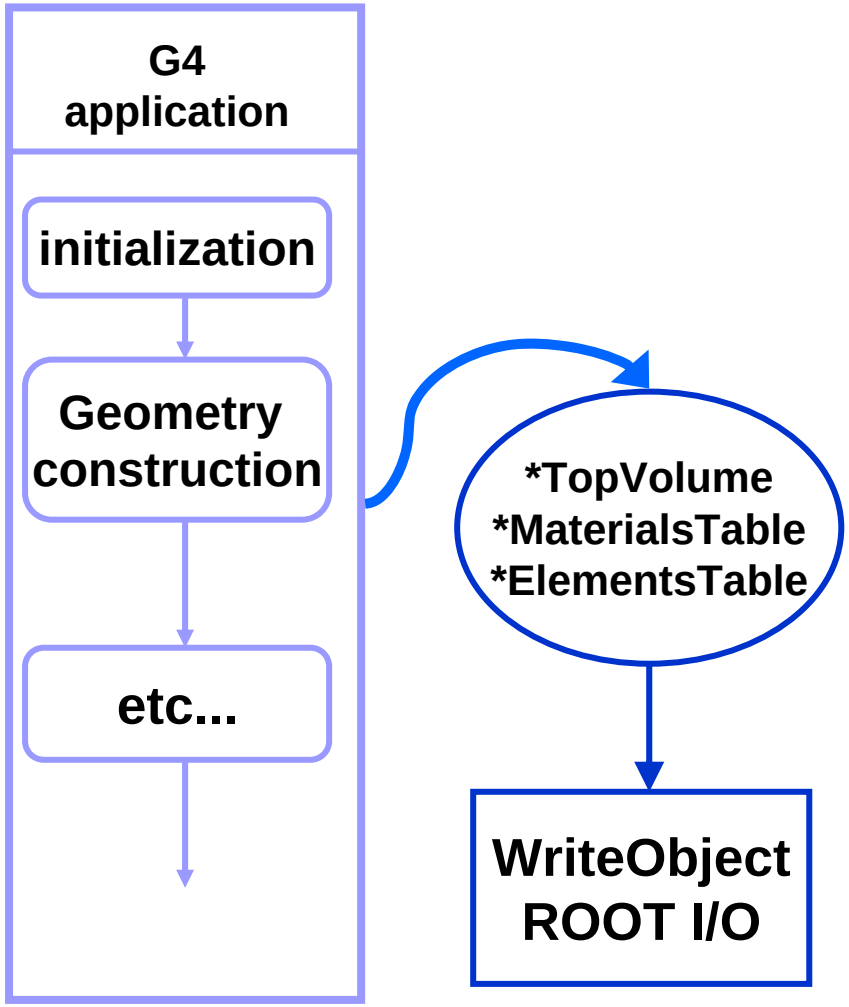
A CAD imported assembly
with tessellated solids - release 8.1

Geometry persistency using ROOT I/O

- Geant4 does not come with any persistency mechanism for the geometry objects
  - the Geant4 geometry tree has to be 'rebuilt' each time
- our goal: to provide a way of quick saving and reading back the G4 geometry in/from a (binary) file using ROOT I/O
  - would nicely extend the functionality of the toolkit
- remark: this is a different use-case from GDML, where universality of the format was top-priority and not the speed

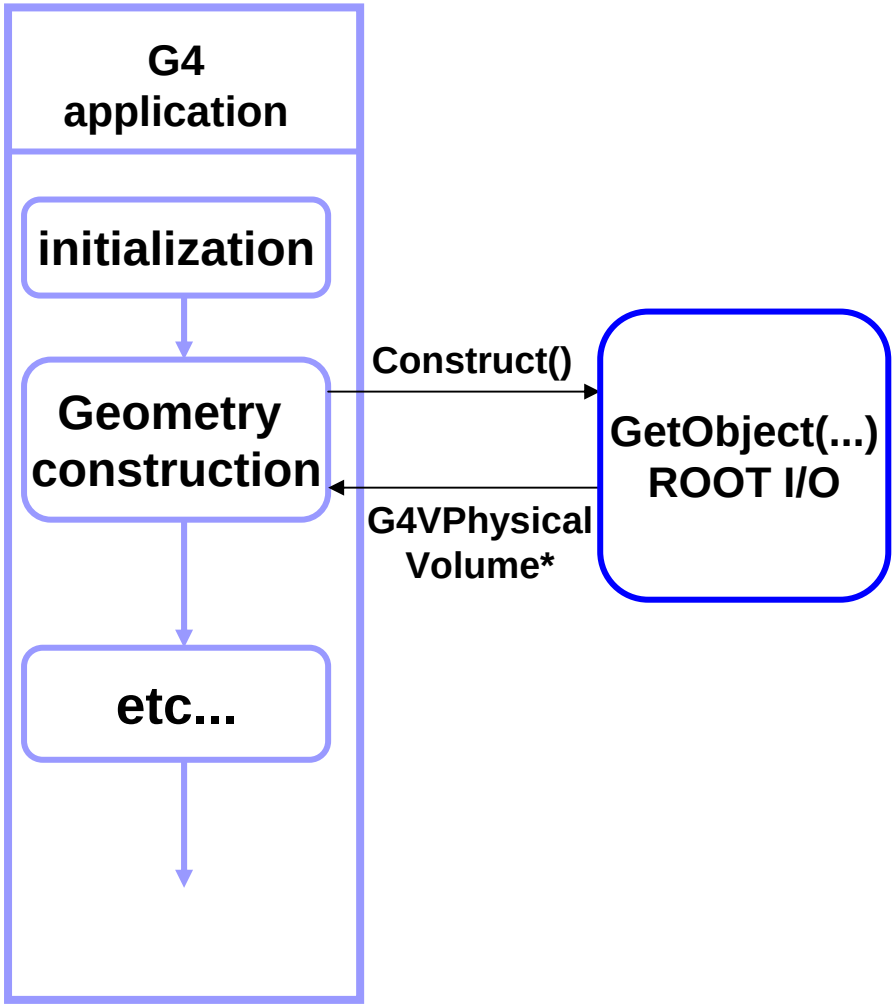
Practical case - writing

- we have our Geant4 geometry in memory and we want to save it in .root file
- we call a simple 'GeoWriter' tool which:
  - creates a 'wrapper' object containing *TopVolume and pointers to materials and elements static tables
  - calls WriteObject ROOT I/O method
- trivial implementation
  - no any 'scanning' of the geometry tree needed
  - ROOT traverses all the geometry tree and stores it
  - only needed thing is to export the pointer to the top volume



Practical case - reading

- only binding to ROOT in DetectorConstruction class
- the Geant4 'main' does not see the loading of the geometry using ROOT I/O
  - 'standard' DetectorConstruction replaced by ROOTDetConstr.
  - G4VUserDetectorConstruction::Construct() returns pointer to the top volume
- ROOTDetectorConstruction as a simple 'plug-in'
  - one just needs to instantiate it from the 'main'



Short-term geometry developments

- Parallel navigation
  - Ability to define multiple navigators for different geometries in parallel (fast-simulation, importance biasing, scoring)
  - Transportation in presence or not of magnetic field
- Tunable tolerance
  - Ability to optionally set tolerance for surface thickness and intersection calculation
  - Automatic evaluation of the tolerance according to the geometry topology (world-volume size)
- Computation of the surface area of a solid
- In plan for coming December release

GDML

GDML - Motivation

- initially developed as an alternative geometry description format for Geant4
  - to move away from hardcoded geometry
    - to allow flexible geometry configuration without the need to recompile
- now, playing also an important role of geometry interchange format
  - application independent
    - same GDML file can be used by several application
  - possibility to export geometries from experiment-specific frameworks
    - allows physics validation/comparison, visualization, debugging

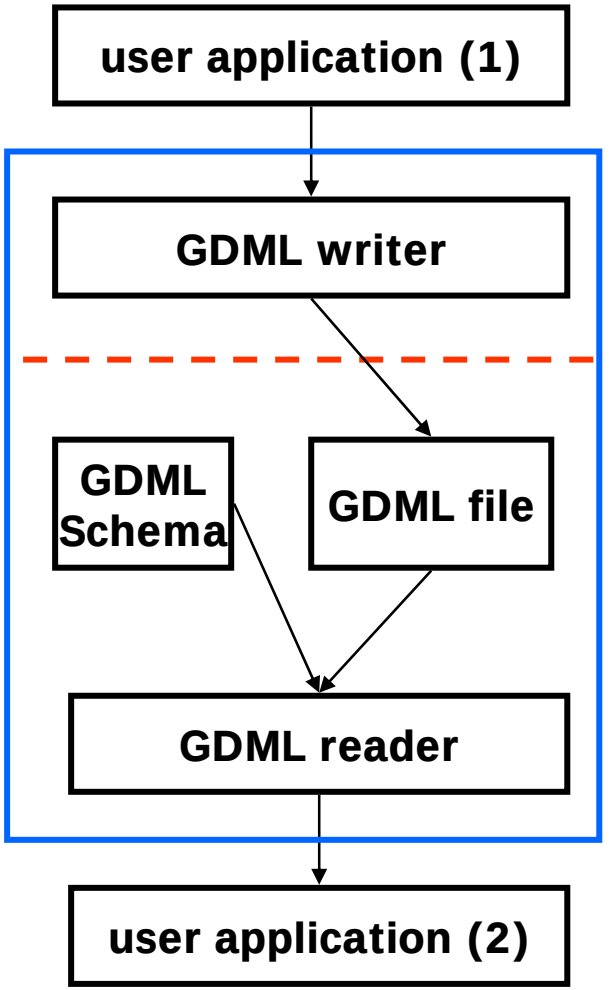
GDML design choice - why XML?

- purpose of GDML is to describe data
  - to provide persistent form of geometry data
  - not procedural, but markup language
- must be easy to read and write
  - no heavy I/O system to read GDML
- format must be application independent
- possibility to edit/read geometry files is an advantage
  - XML file can be edited using any editor
  - geometry can be modified easily
- must be easy to extend and be modular

GDML designed as an application of XML

GDML components

- GDML is defined through XML Schema (XSD)
  - XSD = XML based alternative to Document Type Definition (DTD)
  - defines document structure and the list of legal elements
  - XSD are in XML -> they are extensible
- GDML can be written by hand or generated automatically
  - 'GDML writer' allows writing-out GDML file
- GDML needs 'reader'
  - 'GDML reader' creates 'in-memory' representation of the geometry description



GDML Geant4 binding

- package available from www.cern.ch/gdml
  - latest release GDML_2_8_0 tested with G4.8.1.p01
- autoconf/make based build system
- requires XercesC parser (tested with versions 2.3.0 and 2.7.0)
- could be (in the future) integrated more with G4 distribution
  - optional package to be linked against during build
  - 'GDMLDetectorConstruction provided
  - geometry exportation in GDML steered by UI command?

GDML document

positions, rotations

materials

solids

geometry tree

'world' volume

```
<?xml version="1.0" encoding="UTF-8"?>
<gdml xsi:noNamespaceSchemaLocation="GDMLSchema/gdml.xsd">
    <define>
        …
        <position name="TrackerinWorldpos" unit="mm" x="0" y="0" z="100"/>
    </define>
    <materials>
        …
        <element name="Nitrogen" formula="N"  Z="7.">
        <atom value="14.01"/>  </element>
        <material formula=" " name="Air" >
                <D value="1.290" unit="mg/cm3"/>
                <fraction n="0.7" ref="Nitrogen" />
                <fraction n="0.3" ref="Oxygen" />
        </material>
    </materials>
    <solids>
        …
        <box lunit="mm" name="Tracker" x="50" y="50" z="50"/>
    </solids>
    <structure>
        …
        <volume name="World" >
                <materialref ref="Air" />
                <solidref ref="world" />
                <physvol>
                        <volumeref ref="Tracker" />
                        <positionref ref="TrackerinWorldpos"/>
                        <rotationref ref="TrackerinWorldrot"/>
                </physvol>
        </volume>
    </structure>
    <setup name="Default" version="1.0" >
        <world ref="World" />
    </setup>
</gdml>
```

Materials, solids

- materials
  - material, isotope, element, mixture
- solids
  - all CSG: box, sphere, tube, cone, parallepiped, trapezoid, torus,
  - all specific: polycone, polyhedra, hyperbolic tube, elliptical tube, ellipsoid, elliptical cone, tetrahedron, twisted solids, tessellate solids NEW
  - boolean solids
  - no BREPs
  - single placements, assembly volumes and reflections
  - replicas, divisions, parameterised volumes (first implementation)
- optical surfaces and material property sheets NEW

Loops, matrices NEW

- matrices

```
<matrix name="m"
coldim="3"
values="0.4  9  126
        8.5  7  21
        34.6 7  9" />
```

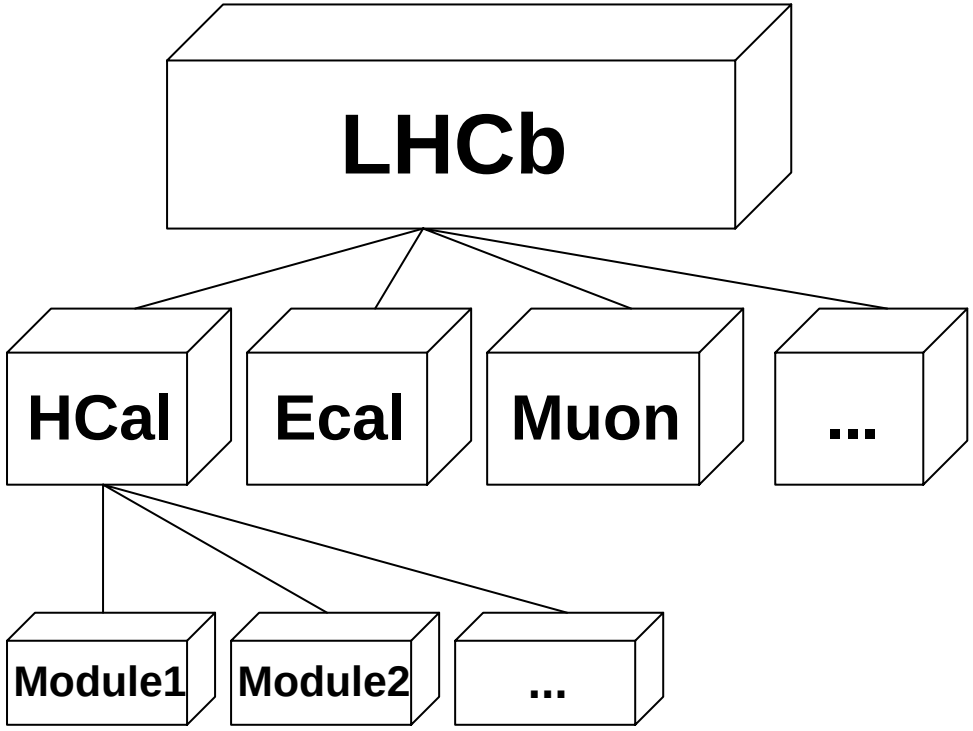
- loops (for multiple placements)

```
<loop for="x" to="5" step="1">
<box name="box" x="10-x" y="5"
z="m[2,x]" />
</loop>
```

- arithmetic expressions allowed (+, -, *, /, sin, cos, etc)
  - evaluated by CLHEP::Evaluator

Modular geometry files NEW

- support for modular GDML geometries
  - several standalone GDML files can now be combined together within another 'top level' GDML file
- GDML writer can now split geometry in modules



Modular description - example

```
<volume name="DetBox" >
   <materialref ref="Air" />
   <solidref ref="detbox" />
   <physvol>
    <volumeref ref="MyVol" />
    <positionref ref="detp0" />
   </physvol>
   <physvol>
    <file name="moduleA.gdml" volname="prova2"/>
    <positionref ref="detp0" />
   </physvol>
</volume>
```

normal physical volume

volume included from file

optional volume name (if not top volume from the file)

Splitting GDML files using ENTITY

- simple mechanism exists to split any XML file in several parts

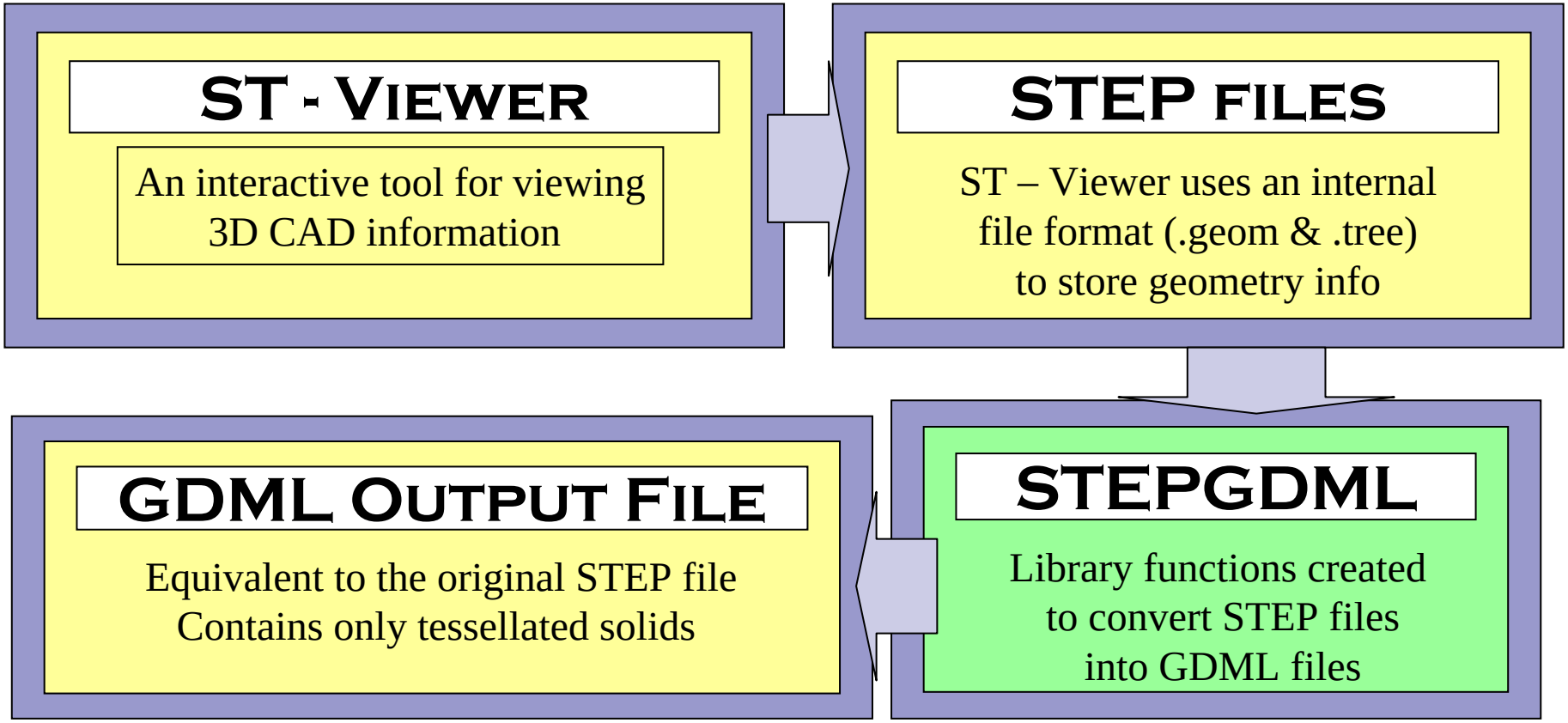
```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE gdml [
<!ENTITY materials SYSTEM "materials.xml">
]>
<gdml ..... >
....
&materials;
....
</gdml>
```

content of materials.xml file will be included here

  - NOTE: the 'extracted' files are not valid GDML files (cannot be used standalone)

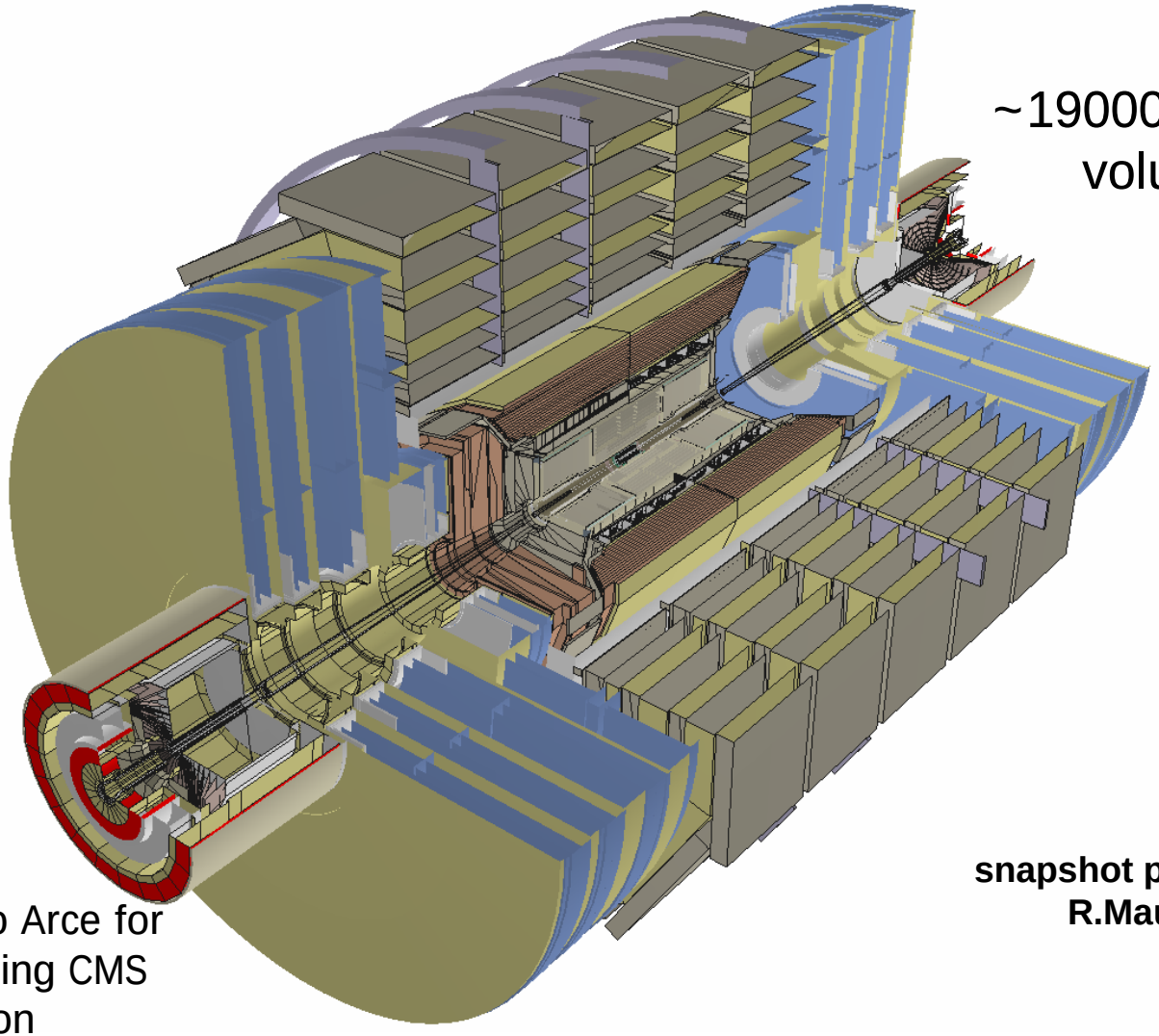
CAD(STEP) -> GDML converter

NEW



- **'first order' approach to use CAD geometries for Geant4 simulation**

CMS detector: G4->GDML->ROOT

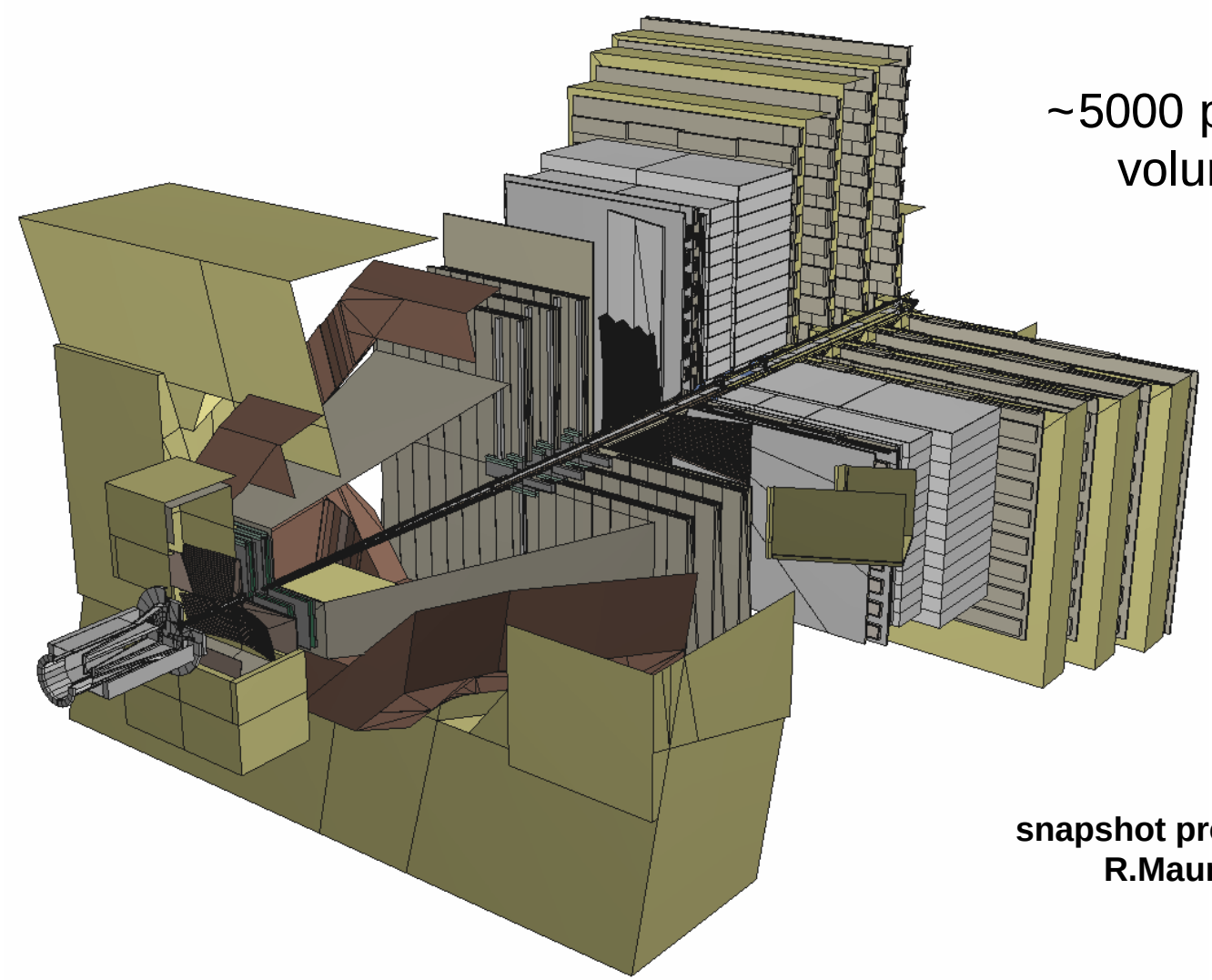


~19000 physical volumes

snapshot provided by R.Maunder

thanks to Pedro Arce for help with running CMS simulation

LHCb Detector: G4->GDML->ROOT



~5000 physical volumes

snapshot provided by R.Maunder

Using GDML with Geant4

to write:

```
#include "WriterG4/G4GDMLWriter.h"
G4GDMLWriter g4writer("GDMLSchema/gdml.xsd", "g4test.gdml");
g4writer.DumpGeometryInfo(g4worldvolume);
```

instantiate GDML writer

pass the 'top' volume to the writer

to read:

```
SAXProcessor sxp;
sxp.Initialize();
ProcessingConfigurator config;
config.SetURI( "g4test.gdml" );
sxp.Configure( &config );

sxp.Run()

fWorld =  (G4VPhysicalVolume *)
            GDMLProcessor::GetInstance()->GetWorldVolume();
```

instantiate and configure the processor

get pointer to 'top' volume

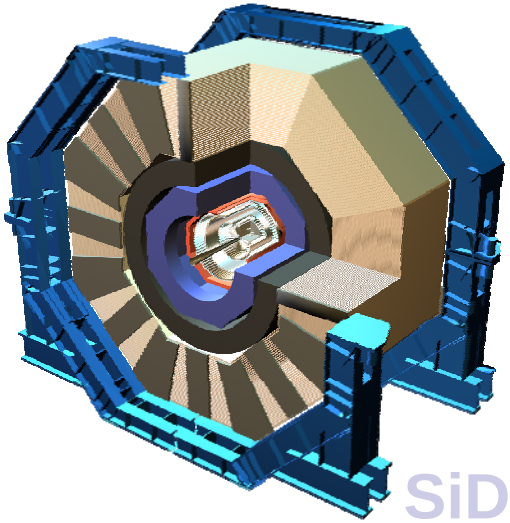
➡ GDML example in examples/extended/gdml

GDML processing - performance

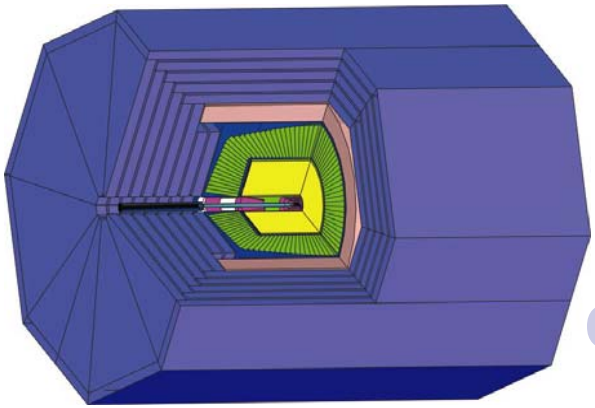
- GDML G4reader/G4writer (C++) tested on
  - complete LHCb and CMS geometries
  - parts of ATLAS geometry
    - problem with full ATLAS geometry - use of custom solids
- for LHCb geometry (~5000 logical volumes)
  - writing out ~10 seconds (on P4 2.4GHz)
  - reading in ~ 5 seconds
  - file size ~2.7 Mb (~40k lines)
- for CMS geometry (~19000 logical volumes)
  - writing out ~30 seconds
  - reading in ~15 seconds
  - file size ~7.9 Mb (~120k lines)

GDML as primary geometry source

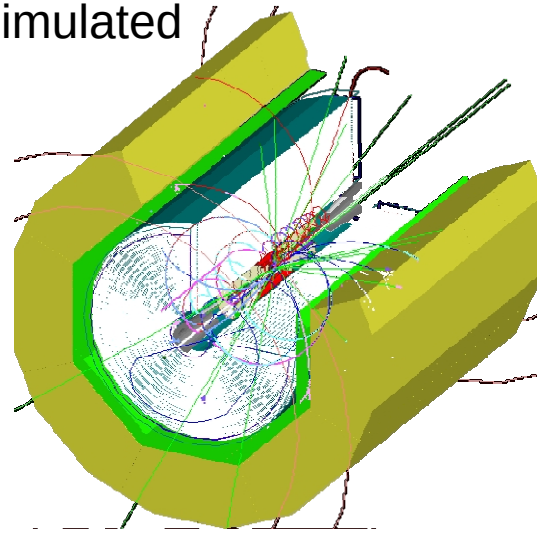
- Linear Collider - Jeremy McCormick, SLAC
  - Linear Collider Detector Description (LCDD) extends GDML with Geant4-specific information (sensitive detectors, physics cuts, etc)
  - GDML/LCDD is generic and flexible
    - several different full detector design concepts, including SiD, GLD, and LDC, where simulated using the same application



SiD



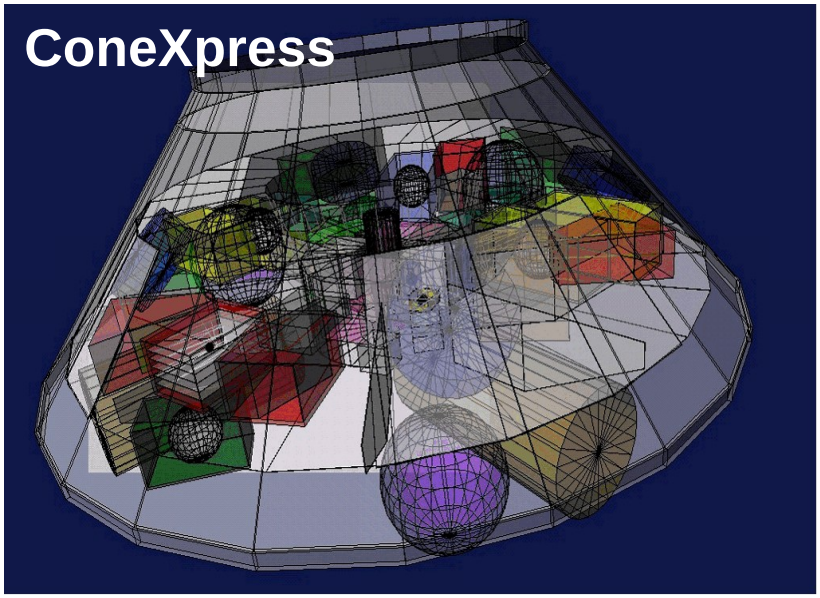
GLD

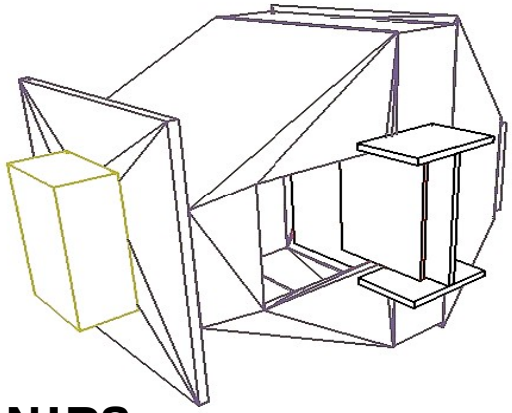


LDC

GDML as primary geometry source

- Space Research - Giovanni Santin, ESA
  - all geometry models for Geant4
    - component degradation studies (JWST, ConeXpress,...)
    - GRAS (Geant4 Radiation Analysis for Space)
  - enables flexible geometry configuration and changes
  - main candidate for CAD to G4 exchange format

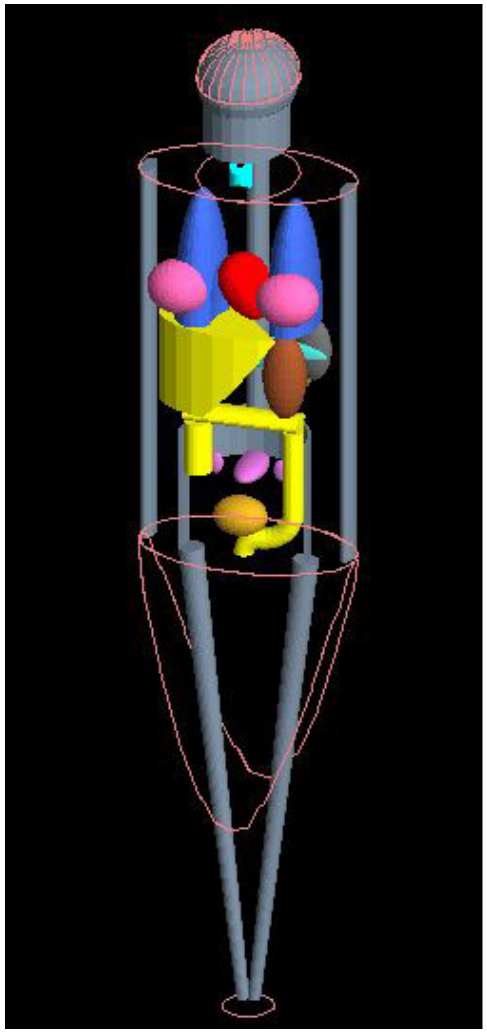




JWST NIRSpec

GDML as primary geometry source

- Anthropomorphic Phantom Project - Giorgio Guerrieri, Maria Grazia Pia, Susanna Guatelli, INFN
  - Modelization of the human body and anatomy for radioprotection studies
  - no hard-coded geometry, flexible configuration



GDML future development

- auxiliary information associated to specific volumes sometimes needed
  - sensitive detector names
  - visualisation attributes
  - magnetic field, etc
- added optional (auxiliary) element to volume:

Will be in next GDML release

```
volume name="VeloSensors" >
    <materialref ref="Silicon" />
    <solidref ref="detectorRUnion" />
    <auxiliary auxtype="sensdet" auxvalue="veloSD"/>
</volume>
```

GDML future development (cont'd)

- <auxiliary...> element has two string attributes
  - parser does not interpret those attributes, only stores them in the map: volume -> (auxtype, auxvalue)
  - user accesses the map to check if for given volume there are any auxiliary attributes
    - example:
      - check if auxiliary attribute with auxtype = "sensdet" exist
      - use G4SDManager::FindSensitiveDetector(auxvalue) to retrieve pointer to appropriate SD
- Jeremy McCormick uses more sophisticated technique, extending volume element and loading his own subscribers
  - this goes it the direction of general GDML extensions by users

GDML - Conclusion

- GDML binding for G4 fully functional
  - example provided in examples/extended/gdml
- can be used as primary format for geometry implementation
- can also play a role of an interchange format