# TGeoCAD: an Interface between ROOT and CAD Systems

ACAT 2013
IHEP - Beijing - China
Author: Cinzia Luzzi

University of Ferrara - CERN

18 May 2013

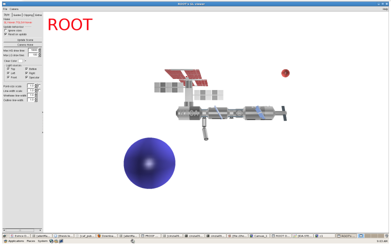
# TGeoCAD

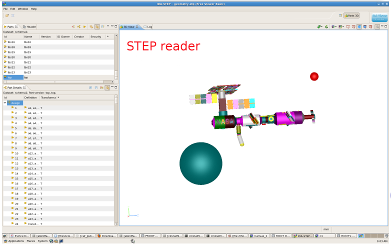


## Purpose

- High precision in the description of the detector geometry is essential;
- Workaround to the incompatibility of software used by physicists and engineers for the simulation and the mechanical design of the detector geometry;

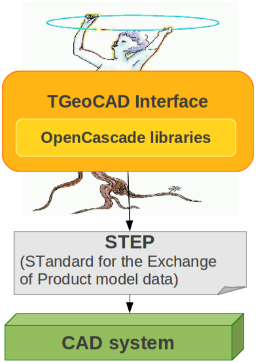
**The TGeoCAD interface enables the use of ROOT files in several CAD systems.**



# TGeoCAD: Components

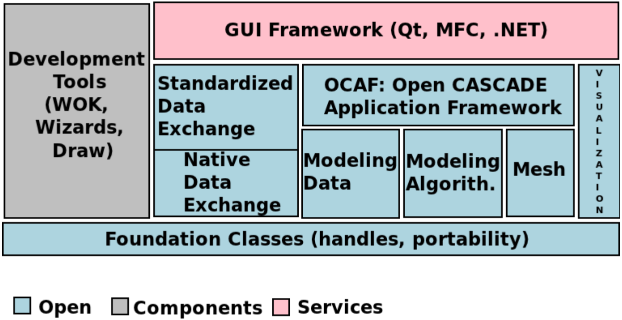
- ROOT
- Open CASCADE Technology (OCCT):
  - Open source software development platform.
  - C++ components for:
    - 3D surface and solid modeling;
    - visualization;
    - data exchange and rapid application development;
- STEP STandard.



# STEP Format (ISO 10303)

- ISO Standard for the Exchange of Product model data.
- Represents 3D objects in Computer-aided design (CAD) and related information.
- The Application Protocols (AP) are the top parts of the STEP Standard;
- OCCT creates files according to STEP AP203 / AP214 parts:
  - Part 203:configuration controlled 3D designs of mechanical parts and assemblies.
  - Part 214:core data for automotive mechanical design processes. It is the default format used by OCCT.
- EXPRESS data modeling language used to describe data models.
- Step-File (Part 21): encoding mechanism on how to represent data according to their EXPRESS schema.

# The OpenCascade Technology

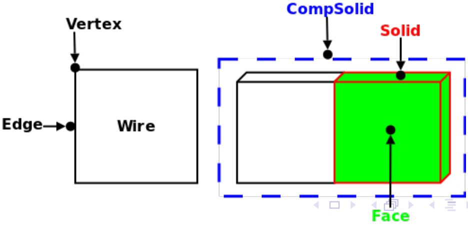


# OpenCascade Foundation Data Toolkit

- Math utilities provides:
  - Description of elementary geometric shapes:
    - a STEP-compliant implementation of basic geometric and algebraic entities;
    - points, vectors, lines, circles and conics, planes and elementary surfaces;
  - Means for positioning geometry in space or on a plane using an axis or a coordinate system;
  - Definition of geometric transformations (translation, rotation and symmetries);

# OpenCascade Modeling Data Toolkit

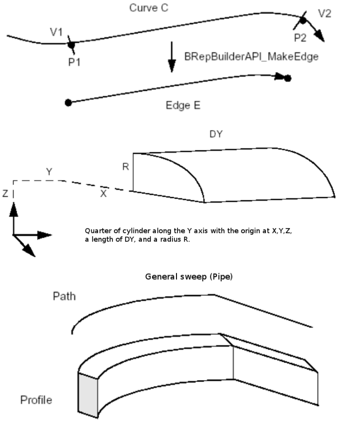
- Data structures to represent 2D and 3D geometric and topological models.
- The topological library allows to build pure topological data structures and to defines relationships between simple geometric entities.
- The abstract topological data structure describes the shape which can be divided into various topological components as shown in the schema:



# OpenCascade Modeling Algorithm Toolkit

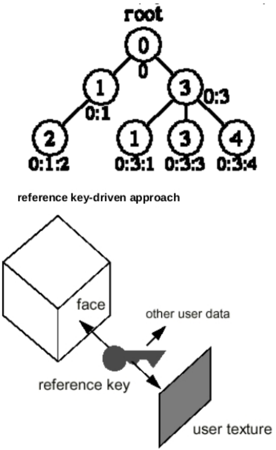


- Geometric and topological algorithms used in modeling:
  - create vertices, edges, faces, solids;
  - build primitive objects (boxes, wedges and rotational objects);
  - perform sweeping operations (Prism - linear sweep, Revolution - rotational sweep and Pipe - general sweep) and boolean operations;



# Open Cascade Application Framework (OCAF)

- Provides an infrastructure to attach any data to any topological element;
- Application/document architecture:
  - Data structure is reference-key driven;
  - The reference key is implemented in the form of labels.
  - Application data is attached to these labels as attributes (shape, general, relationship etc).
  - The set of labels organized in a tree structure is kept in the document. Each label has a tag expressed as an integer value.
  - A label is a string build by concatenation of tags from the root of the tree, for example [0:1:2].



# Data Exchange and Extended Data Exchange (XDE) Toolkits



- It gives the possibility to write the OCAF document in a STEP file;
  - it allows to the software based on Open Cascade to exchange data with various CAD software;
- The labels tree structure becomes for XDE an assembly structure composed by several components (shape, subshape):
  - The location of a shape can be defined as attribute.
  - The same shape can be used several times in the assembly structure redefining the location.

# TGeoCAD Geometry Conversion

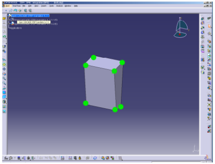
- Shapes created step by step starting from points (edge, wire, face, shell and solid) such as:
  - Box;
  - Parallelepiped;
  - Trapezoid;
- Shapes created using OCCT capabilities for solid primitives creation and boolean operations such as:
  - Tube;
  - Cone;
  - Sphere;
- Shapes created by using modeling algorithm (extrusions, revolutions, lofts) applied to basic geometries such as:
  - Hyperboloid;
- TGeoCompositeShape created using OCCT boolean operations between two or more shapes;

# TGeoCAD Geometry Conversion



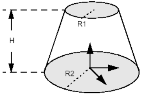
## TGeoTrd1

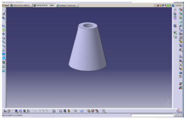
Creates edges from points, wire from edges, faces (planar surfaces) from wires, shells from faces and solid from shells;



## TGeoCone

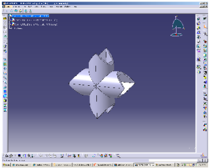
Starting from radius, creates inner and outer cones;
Substract the inner cone from the outer cone.



## TGeoCompositeShape

Boolean operations between a box, a tube and a pgon.



# TGeoCAD Classes

## TGeoToStep

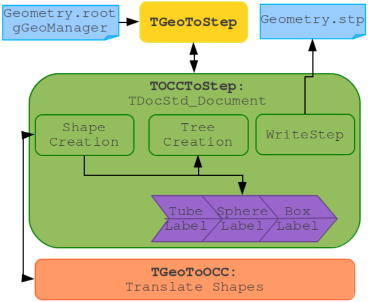
Takes a gGeoManager pointer and gives back a STEP file;

## TGeoToOCC

For each ROOT shape creates the corresponding OCCT shape.

## TOCCToStep

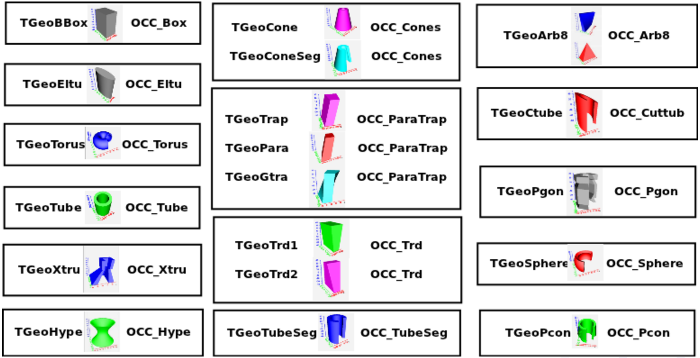
Reproduces the ROOT tree (mother-children relationship) on the XDE document and writes it to the STEP file;



# TGeoToOCC Class

## Converts ROOT shapes to OCCT shapes.

Each ROOT shape is translated in the correspondent OCC shape using the following methods:



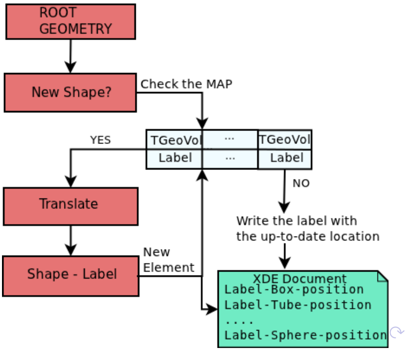
# TOCCToStep Class



## TOCCToStep::OCCShapeCreation

Starting from the top of the ROOT geometry tree translates each ROOT shape in the OCCT version.

- For each shape a new label is written in the XDE document. The correspondance shape-label is stored in a map of volumes and labels;
- If the shape-label correspondance is present in the map, add the label to the document updating the location;
- A shape positioned several times in a ROOT volume is translated only once;



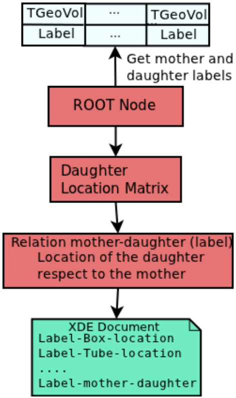
# TOCCToStep Class



## TOCCToStep::OCCTreeCreation

For each node from the end to the top of the ROOT physical tree:

- Gets mother and daughter label reference from the map;
- Takes the daughter location matrix;
- Connects the daughter label to the mother label with its location resulting in a new label;
- Adds the new label to the document which reproduce the relationship mother-children.



# TOCCToStep Class

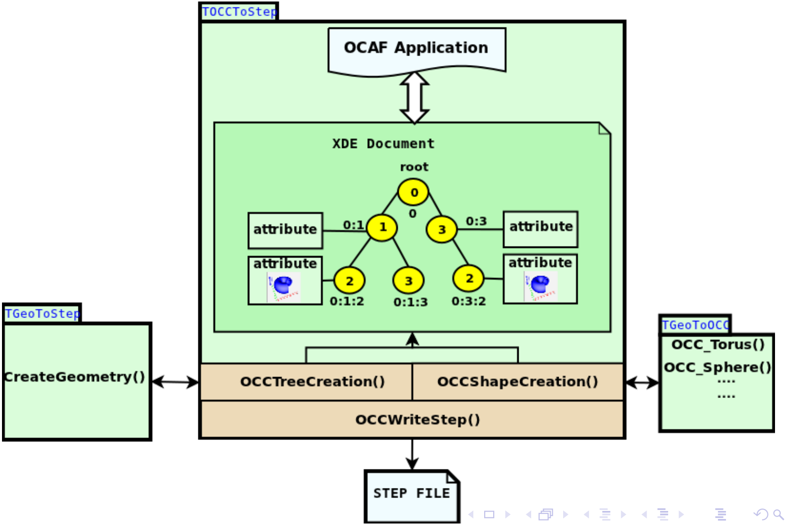
## TOCCToStep::OCCWriteStep

Writes the XDE document on the step file using STEPCAFControl_Writer OCC class;

```
3103,
  'distance_accuracy_value','confusion accuracy');
#3107 = SHAPE_DEFINITION_REPRESENTATION(#3108,#2783);
#3108 = PRODUCT_DEFINITION_SHAPE('','',#3109);
#3109 = PRODUCT_DEFINITION('design','',#3110,#3113);
#3110 = PRODUCT_DEFINITION_FORMATION('','',#3111);
#3111 = PRODUCT('bar1','bar1','',(#3112));
#3112 = MECHANICAL_CONTEXT('',#2,'mechanical');
#3113 = PRODUCT_DEFINITION_CONTEXT('part definition',#
2,'design');
#3114 = CONTEXT_DEPENDENT_SHAPE_REPRESENTATION(#3115,#3117);
#3115 = ( REPRESENTATION_RELATIONSHIP('','',#2783,#1551)
REPRESENTATION_RELATIONSHIP_WITH_TRANSFORMATION(#3116)
SHAPE_REPRESENTATION_RELATIONSHIP() );
#3116 = ITEM_DEFINED_TRANSFORMATION('','',#11,#1568);
#3117 = PRODUCT_DEFINITION_SHAPE('Placement','Placement of an
item',
  #3118);
#3118 = NEXT_ASSEMBLY_USAGE_OCCURRENCE('14','=>[0:1:1:5]','',#
1546,#3109
  ,$);
#3119 = PRODUCT_TYPE('part',$,(#3111));
#3120 = CONTEXT_DEPENDENT_SHAPE_REPRESENTATION(#3121,#3123);
#3121 = ( REPRESENTATION_RELATIONSHIP('','',#1551,#81)
REPRESENTATION_RELATIONSHIP_WITH_TRANSFORMATION(#3122)
SHAPE_REPRESENTATION_RELATIONSHIP() );
#3122 = ITEM_DEFINED_TRANSFORMATION('','',#11,#94);
#3123 = PRODUCT_DEFINITION_SHAPE('Placement','Placement of an
```

Relationship definition between shape label 2783 and 1551

# Summary Schema



# TGeoCAD Requirements

- OCCT must be installed (export CASROOT=path-to-OCCT);
- ROOT must be compiled using the configuration options:

```
svn co http://root.cern.ch/svn/root/trunk root
./configure --enable-geocad;
--with-occ-incdir:  location of OpenCascade inc files
($CASROOT/inc);
--with-occ-libdir:  location of OpenCascade lib files
($CASROOT/lib);
```

- A ROOT geometry must be loaded in the memory.

```
root[0] gSystem->Load("libGeoCad.so");
root[1] .x roottest.C
root[2] TGeoToStep *myStep = new TGeoToStep (gGeoManager);
root[3] myStep->CreateGeometry();
```

Thank you for your attention!