The G4MICE Geometry Interface to the Configuration Database

Ivan D Reid

Brunel University

Goal

- To be able to save a MiceModule geometry description tree to the Configurations Data-Base and to retrieve the data to reconstruct the tree
- Storage to be done in XML

Design Decision

- The existing Geometry Description Markup Language (GDML) was chosen

  http://lcgapp.cern.ch/project/simu/framework/GDML/gdml.html
- Attempts to translate directly into valid GDML were hampered by some mis-matches between GDML and MiceModule (e.g. scale, available volume types)
- Design was changed to store MiceModule data as volume “**auxiliary**” data (strings)

Schema Translation

- The GDML_3.0.0 schema was translated into C++ classes using C++/Tree (xsd) from codesynthesis

  http://www.codesynthesis.com/products/xsd/c++/tree/
- C++/Tree uses the Xerces-C++ library V3.1.1 (which was built from sources)
- xsd and xerces-c are also available in SLC5 and ubuntu repositories

Pseudo-GDML

- The resultant “pseudo-GDML” looks like:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<gdml version="2.10.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://service-spi.web.cern.ch/service-spi/app/releases/GDML/GDML_3_0_0/schema/gdml.xsd">

  <define>
    <constant name="Creation_Time_UTC" value="2011-01-20 16:28:58"/>
    <position name="DefaultPosition" x="0" y="0" z="0"/>
    <rotation name="DefaultRotation" x="0" y="0" z="0"/>
  </define>

  <materials>
    <element Z="1" formula="H" name="Hydrogen" state="unknown">
      <atom value="1.01"/>
    </element>
    <material name="Default" state="unknown">
      <D value="1.e-25"/>
      <composite n="1" ref="Hydrogen"/>
    </material>
  </materials>

  <solids>
    <box aunit="radian" lunit="mm" name="None" x="1" y="1" z="1"/>
    <box aunit="radian" lunit="mm" name="MICE" x="6000" y="6000" z="31000"/>
  </solids>

  <structure>
…
```

...continued

```
 <volume name="Stage6.datStage6CoilCurrents.datCouplingCoil123_vol">
      <materialref ref="Default"/>
      <solidref ref="None"/>
      <auxiliary auxtype="str_Name" auxvalue="CouplingCoil123"/>
      <auxiliary auxtype="bool_RepeatModule" auxvalue="T"/>
      <auxiliary auxtype="int_NumberOfRepeats" auxvalue="1"/>
      <auxiliary auxtype="dbl_CurrentDensity" auxvalue="1"/>
      <auxiliary auxtype="dbl_InnerRadius" auxvalue="725"/>
      <auxiliary auxtype="dbl_Length" auxvalue="250"/>
      <auxiliary auxtype="dbl_RepeatScaleFactor" auxvalue="-1"/>
      <auxiliary auxtype="dbl_ScaleFactor" auxvalue="-96.21"/>
      <auxiliary auxtype="dbl_Thickness" auxvalue="116"/>
      <auxiliary auxtype="str_FieldType" auxvalue="Solenoid"/>
      <auxiliary auxtype="str_FileName" auxvalue="coupling.fld"/>
      <auxiliary auxtype="str_Volume" auxvalue="None"/>
      <auxiliary auxtype="vec_Position" auxvalue="0 0 -1375"/>
      <auxiliary auxtype="vec_RepeatRotation" auxvalue="0 0 0"/>
      <auxiliary auxtype="vec_RepeatTranslation" auxvalue="0 0 2750"/>
      <auxiliary auxtype="vec_Rotation" auxvalue="0 0 0"/>
</volume>
<volume name="World">
      <materialref ref="Default"/>
      <solidref ref="MICE"/>
<physvol>
        <volumeref ref="Stage6.datStage6CoilCurrents.dat_vol"/>
        <positionref ref="DefaultPosition"/>
        <rotationref ref="DefaultRotation"/>
</physvol>
      <auxiliary auxtype="str_Name" auxvalue="Stage6.dat"/>
      <auxiliary auxtype="dbl_G4StepMax" auxvalue="1"/>
      <auxiliary auxtype="str_Material" auxvalue="AIR"/>
      <auxiliary auxtype="str_Volume" auxvalue="Box"/>
      <auxiliary auxtype="vec_Dimensions" auxvalue="6000 6000 31000"/>
    </volume>
  </structure>

  <setup name="Stage6.dat" version="1.0">
    <world ref="World"/>
  </setup>
</gdml>
```

Getting it into the DB

- The ConfigDB presents an interface schema to the Web as a **wsdl** object
- The schema is converted to a C++ interface using **wsdl2h** and **soapcpp2**, both elements of **gSOAP** from **genivia.com**

  http://www.genivia.com/Products/downloads.html
- gsoap is also in SLC5 and ubuntu reps

>Squeezing it<

- Because of the highly repetetive nature of the pseudo-GDML, it compresses well
- However, lightweight compression libs for C++ streams are scarce. One such is **`xstream`** http://xstream.sourceforge.net/
- This was built from source; it needed an include of **`<iostream>`** in one module and mods to Makefile to install headers correctly

Result

- The current code produces 762 KiB GDML strings from the test MiceModule tree; the bz2 compression in `xstream` reduces this to 12 KiB of binary data for transfer to/from the ConfigDB and storage therein
- Repeated tests have shown identical MMTrees from input cards and from GDML at every stage before/after compression and/or transfer to/from CDB.
- System tests are under development

Other URLs

- Geometry API: http://hepunx.rl.ac.uk/egee/mice/doc/
- More info on GDML+MICE: http://micewww.pp.rl.ac.uk:8080/projects/g4mice/wiki/GDML
- GDML C++ interface class members: http://www.mice.iit.edu/software/doxygen/html/
- MiceModuleConfigDBIO class: should be in doxygen; perhaps my commenting is wrong…

Public Methods

```
class MiceModuleConfigDBIO {
public:
MiceModuleConfigDBIO();
virtual ~MiceModuleConfigDBIO() {}
void WriteGDML(std::ostream& Output);
void ReadGDML(std::istream& Input);
bool GetCDBStatus(std::string & aStatus);
bool SaveGDML(std::string & IOVTime);
bool GetCurrentGDML();
std::string GetGeometryIDs();
std::string GetGeometryIDs(std::string * IOVTime);
std::string GetGeometryIDs(std::string * IOVTime1,
                           std::string * IOVTime2);
bool GetGeometryForID(std::string & ID);
bool GetGeometryForRun(std::string & Run);
std::string NowTime();
...
```

Problems

- C++/Tree generated classes include lots of common code(?) – .o files are O(1MB) → lib file ~100MB but applications are 4-5MB
- Missing include in xstream code (bug reported 12/2008) + error in `make install` (bug report filed 15/2 by micewrangler ☺)
- Makefiles are a mess!

Breaking News

- A DumpMiceModule method added to MiceModules so the entire tree can be written as a single MM file for use when ConfigDB is unavailable, or as a basis for experimentation
- Applications are being developed to write Geometries to the DB, list GDML IDs, etc.

Backup

From this (**MiceModule**): To this (**GDML**):

```
Module Stage6CoilCurrents {
 Volume None
 Module CouplingCoil {
  Volume None
  Position 0.0 0.0 -1375.0 mm
  Rotation 0.0 0.0 0.0 degree
  ScaleFactor -96.21
  PropertyString FieldType Solenoid
  PropertyString FileName coupling.fld
  PropertyDouble CurrentDensity 1.
  PropertyDouble Length 250.
  PropertyDouble Thickness 116.
  PropertyDouble InnerRadius 725.
  PropertyBool RepeatModule 1
  PropertyInt NumberOfRepeats 1
  PropertyHep3Vector RepeatTranslation 0.0 0.0 2.75 m
  PropertyHep3Vector RepeatRotation 0.0 0.0 0.0
    degree
  PropertyDouble RepeatScaleFactor -1.
} …
```

```
<volume
     name="Stage6.datStage6CoilCurrents.datCouplingCoil123_vol">
  <materialref ref="Default"/>
  <solidref ref="None"/>
  <auxiliary auxtype="str_Name" auxvalue="CouplingCoil123"/>
  <auxiliary auxtype="bool_RepeatModule" auxvalue="T"/>
  <auxiliary auxtype="int_NumberOfRepeats" auxvalue="1"/>
  <auxiliary auxtype="dbl_CurrentDensity" auxvalue="1"/>
  <auxiliary auxtype="dbl_InnerRadius" auxvalue="725"/>
  <auxiliary auxtype="dbl_Length" auxvalue="250"/>
  <auxiliary auxtype="dbl_RepeatScaleFactor" auxvalue="-1"/>
  <auxiliary auxtype="dbl_ScaleFactor" auxvalue="-96.21"/>
  <auxiliary auxtype="dbl_Thickness" auxvalue="116"/>
  <auxiliary auxtype="str_FieldType" auxvalue="Solenoid"/>
  <auxiliary auxtype="str_FileName" auxvalue="coupling.fld"/>
  <auxiliary auxtype="str_Volume" auxvalue="None"/>
  <auxiliary auxtype="vec_Position" auxvalue="0 0 -1375"/>
  <auxiliary auxtype="vec_RepeatRotation" auxvalue="0 0 0"/>
  <auxiliary auxtype="vec_RepeatTranslation" auxvalue="0 0 2750"/>
  <auxiliary auxtype="vec_Rotation" auxvalue="0 0 0"/>
</volume>
…
```