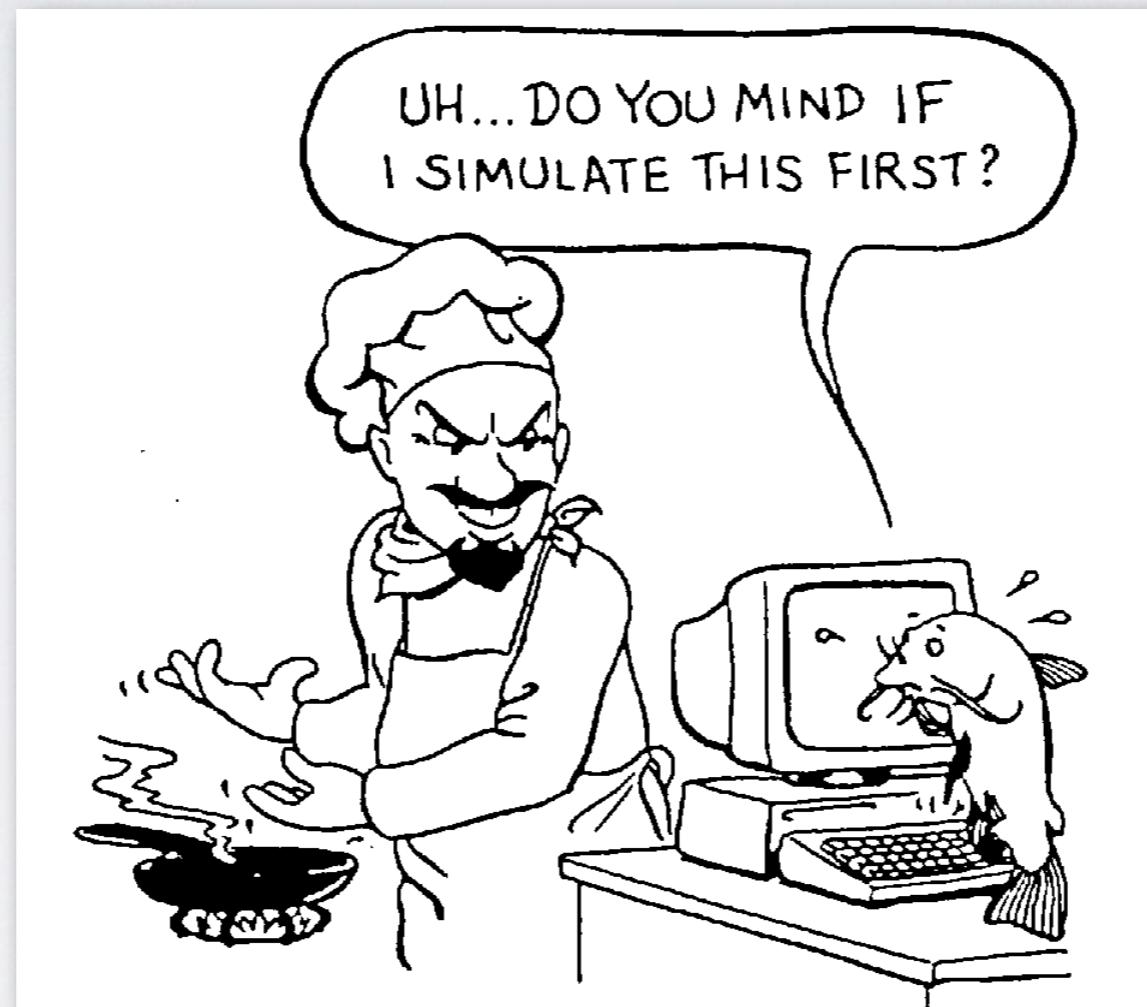
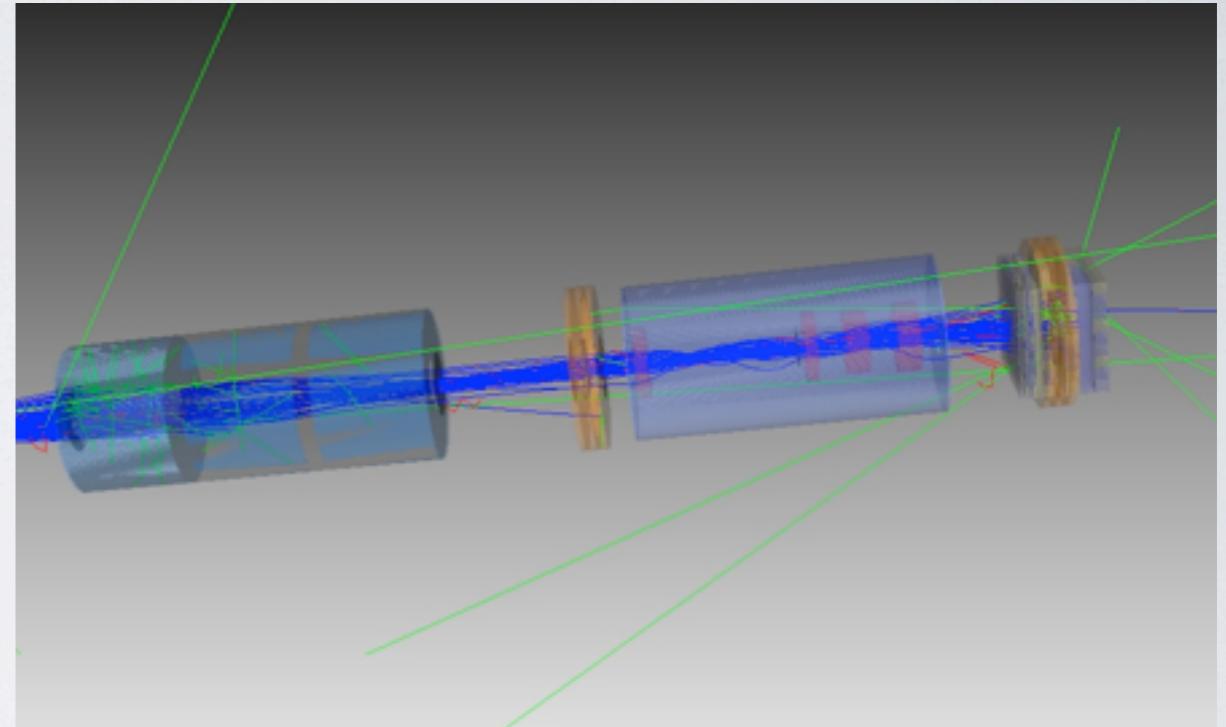


GEMC

GEANT4 MONTE-CARLO



GEANT4 Library



Beam in fields, materials

- Particles through matter
- Several physics environment
- (Electro)Magnetic Fields
- Geometry Navigation

GEANT4 is a C++ Library

Does require some programming language experience

 Does require some C++ knowledge

 OO Programming skills much preferable

GEANT4 is a C++ Library

Does require some programming language experience

 Does require some C++ knowledge

 OO Programming skills much preferable

Hmm...

GEANT4 is a C++ Library

```
G4Box(const G4String& pName,
      G4double  px,
      G4double  py,
      G4double  pz)
```

[Rotate the Picture]
In the picture:
px = 30, py = 40, pz = 60

by giving the box a name and its half-lengths along the X, Y and Z axis:

px|half length in X|py|half length in Y|pz|half length in Z|

Write Geometry in the code

Write Response Function in the code

Implement Magnetic Fields in the code

Incorporate Digitization

Input and Output

Do this for hundreds (thousands) of detector elements

GEANT4 is a C++ Library

```
G4Box(const G4String& pName,
      G4double  pX,
      G4double  pY,
      G4double  pZ)
```

[Rotate the Picture]
In the picture:
px = 30, py = 40, pz = 60

by giving the box a name and its half-lengths along the X, Y and Z axis:

px	half length in X	py	half length in Y	pz	half length in Z
----	------------------	----	------------------	----	------------------

Solid

Logical

Physical

Sensitivity

Fields

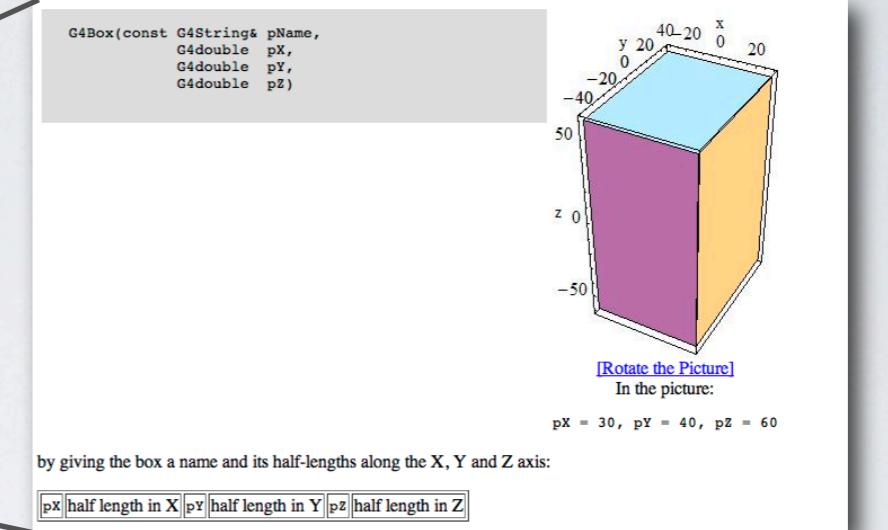
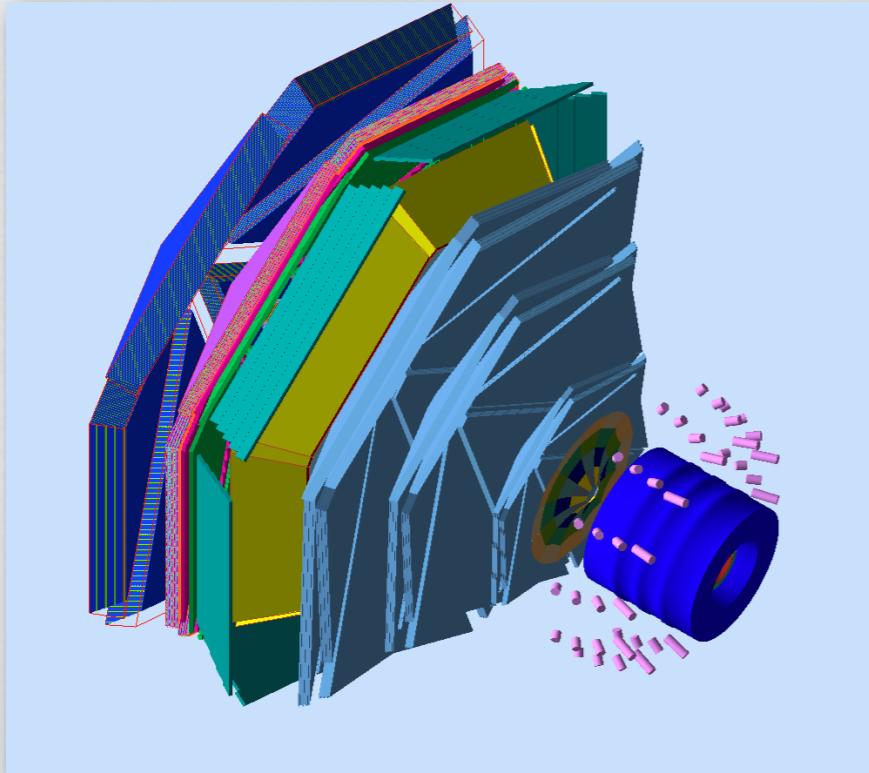
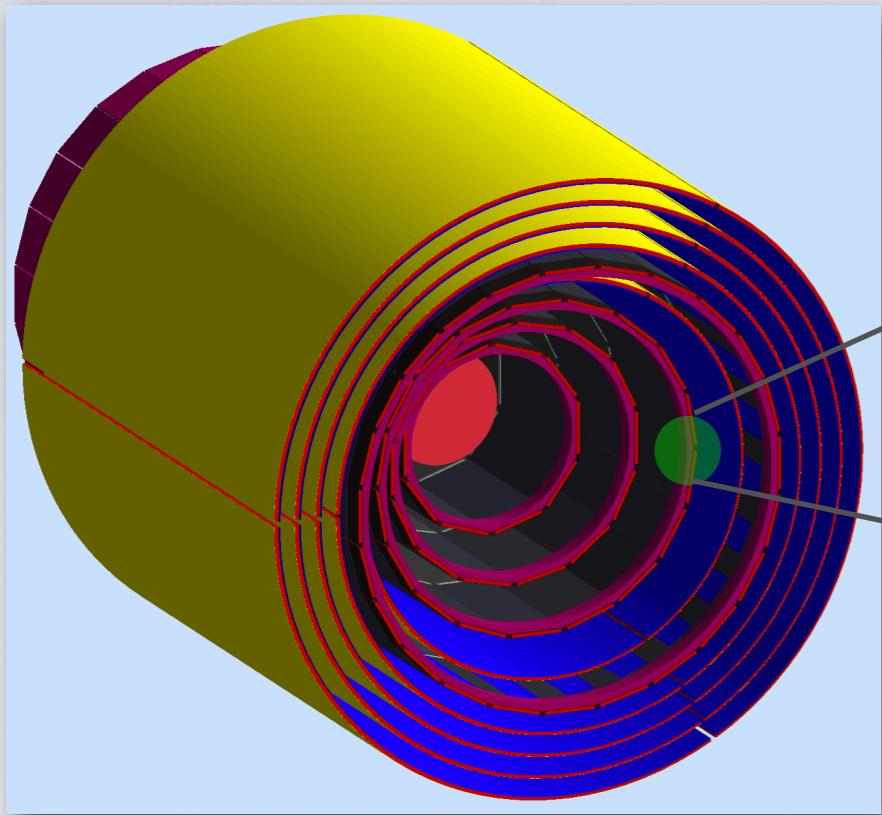
IN

RUN

Out

These steps are well defined

Realistic Simulation



Thousands of detector elements

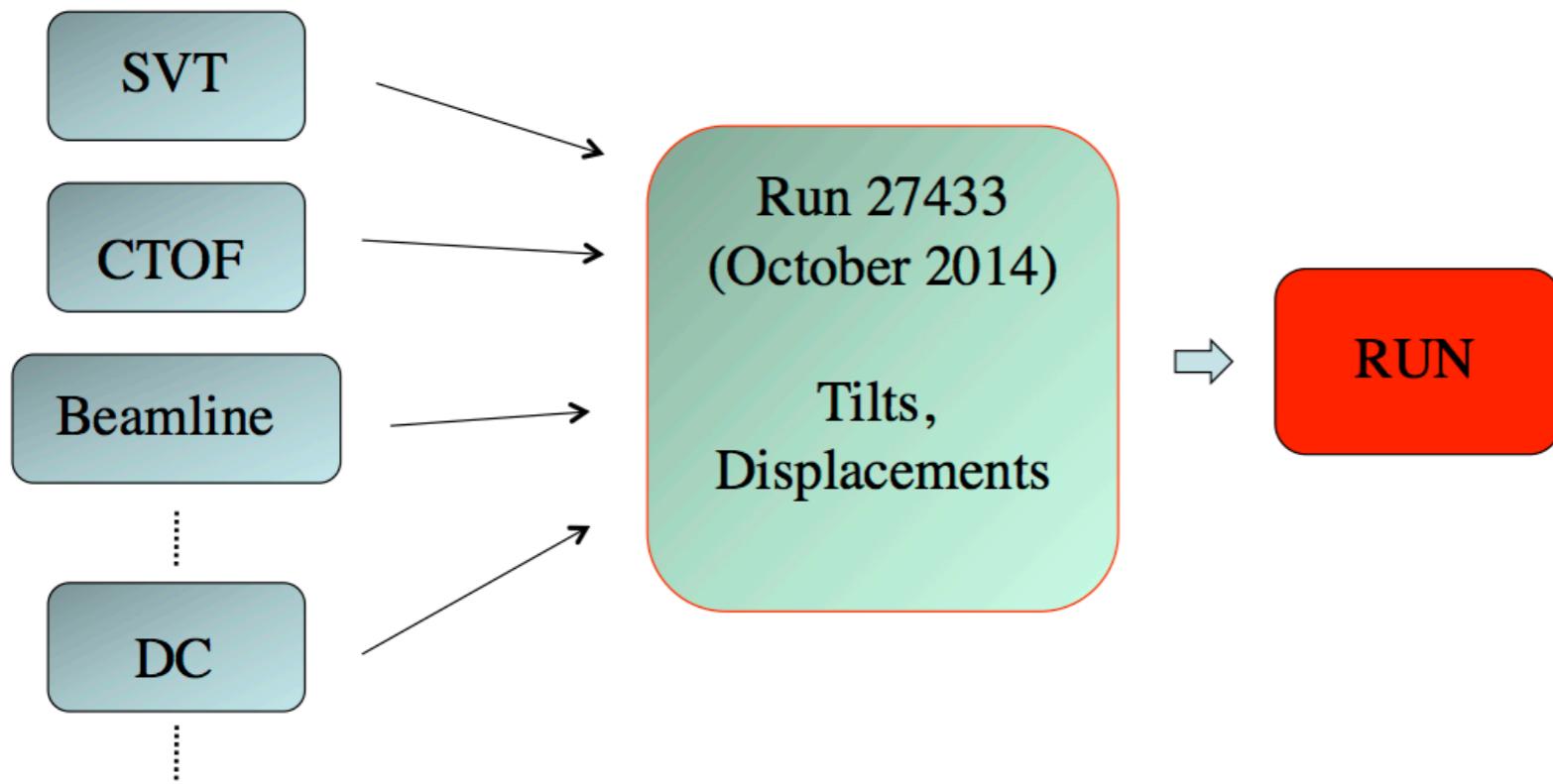
Geometry, digitization Forever Changing

Development, Distribution Nightmare

Copy all parameters to Reconstruction, Event Display

What is an IDEAL simulation environment

Detector, Run (time) - based



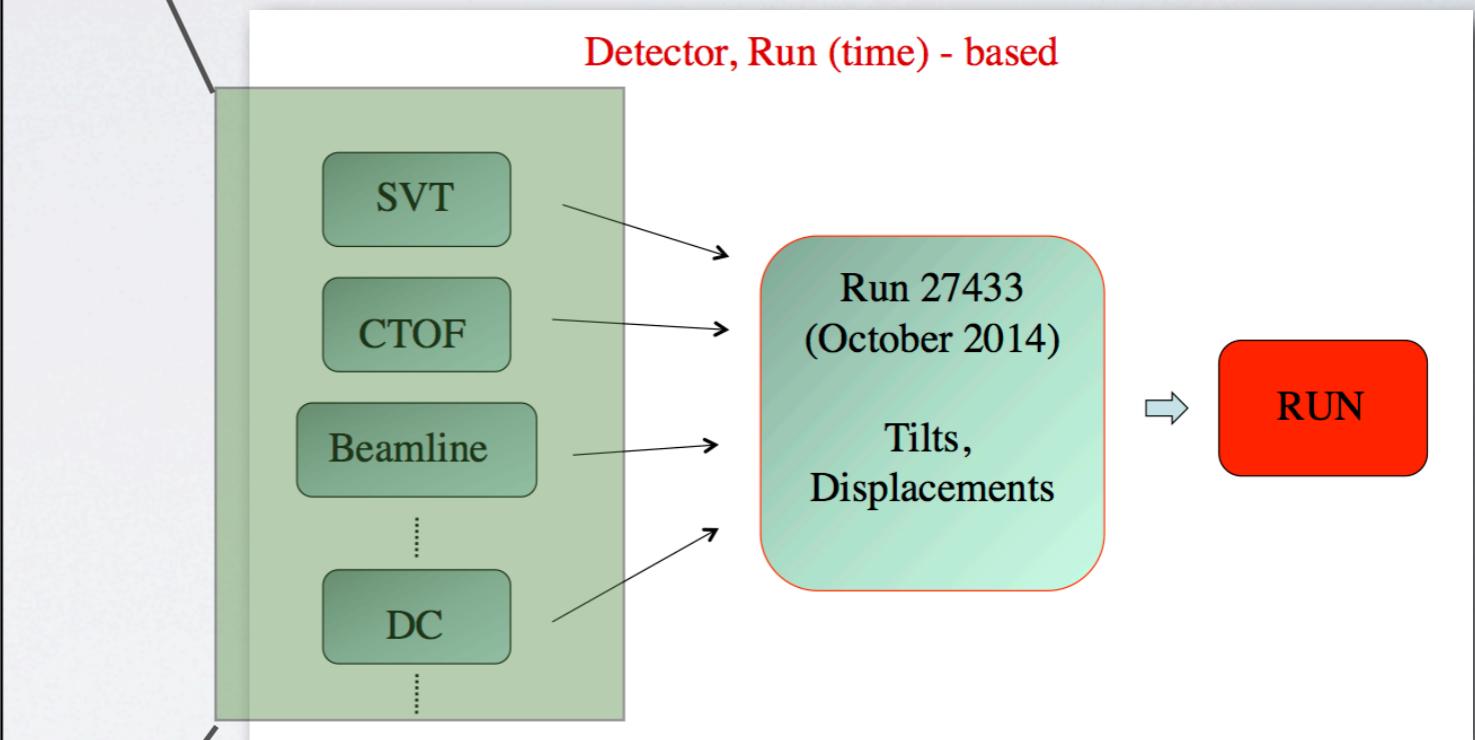
We want to solve these problems:

1. Users should have to deal ONLY with geometry, fields, response function parameters - NOT how they interface with geant4 or the C++ software
2. All parameters should be in a database in common with reconstruction, calibration, visualization

GEant4 Monte-Carlo

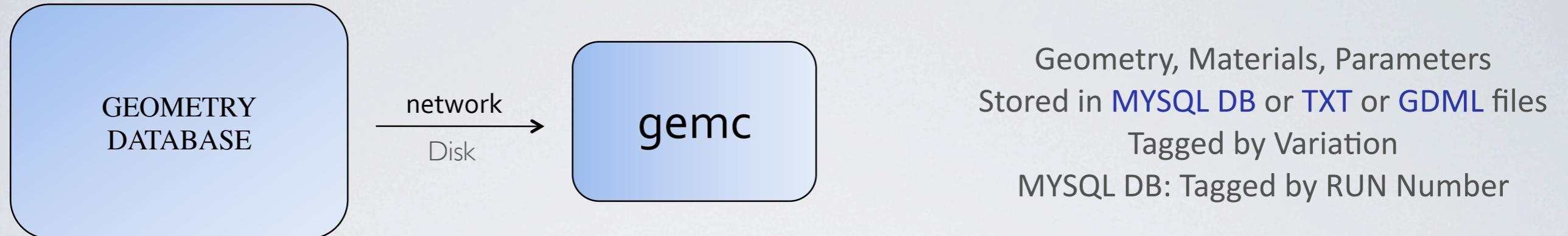
**Database
MYSQL,
GDML,
TXT,**

The Geometry
Sensitivity
Digitization
Output Format
Magnetic Fields
Materials



Tagged by RUN NUMBER, VARIATION

GEMC Geometry Factories



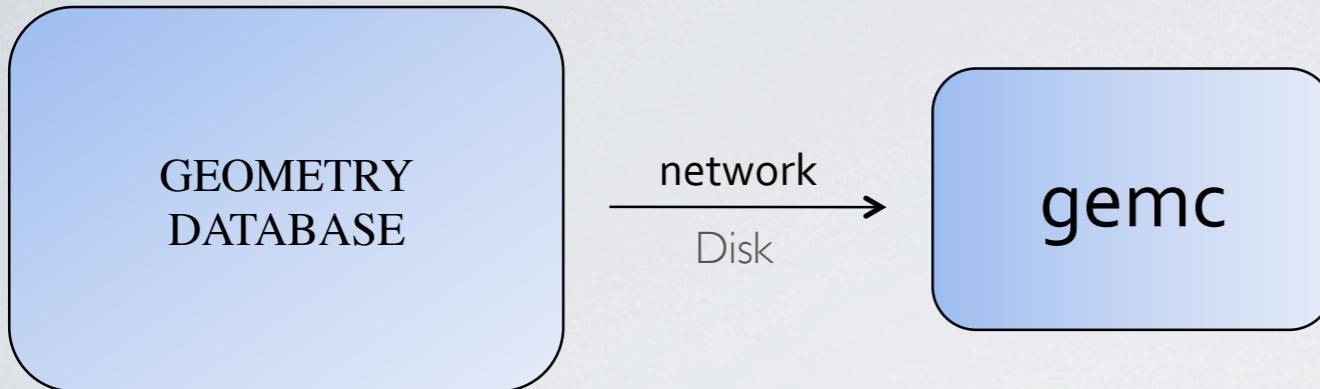
```
<gcard>
```

```
  <detector name="tagger" factory="GDML" variation="maurizio_1" />
  <detector name="tof" factory="TEXT" variation="maurizio_1" />
```

```
</gcard>
```

Offline Mode

GEMC Geometry Factories



Geometry, Materials, Parameters
Stored in **MYSQL DB** or **TXT** or **GDML** files
Tagged by Variation
MYSQL DB: Tagged by RUN Number

```
<gcard>

<option name="DBHOST"    value="clasdb.jlab.org"/>      < can be any server, localhost
<option name="DATABASE" value="clas12"/>

<detector name="beamline" factory="MYSQL" variation="original" r="22"/>
<detector name="drift chambers" factory="MYSQL" variation="Mac_Mesta" r=22" />

<detector name="tagger" factory="GDML" variation="maurizio_1"  />
<detector name="tof" factory="TEXT" variation="maurizio_1"  />

</gcard>
```

GEMC DB I/O

```
sub make_ITOF
{
    for(my $n=1; $n<=$NUM_BARS; $n++)
    {

        # positioning
        # The angle $theta is defined off the y-axis (going clockwise) so $x and $y are reversed
        my $theta = ($n-1)*$theta0;
        my $theta2 = $theta + 90;
        my $x     = sprintf("%.3f", $R*cos(rad($theta)));
        my $y     = sprintf("%.3f", $R*sin(rad($theta)));
        my $z     = "0";

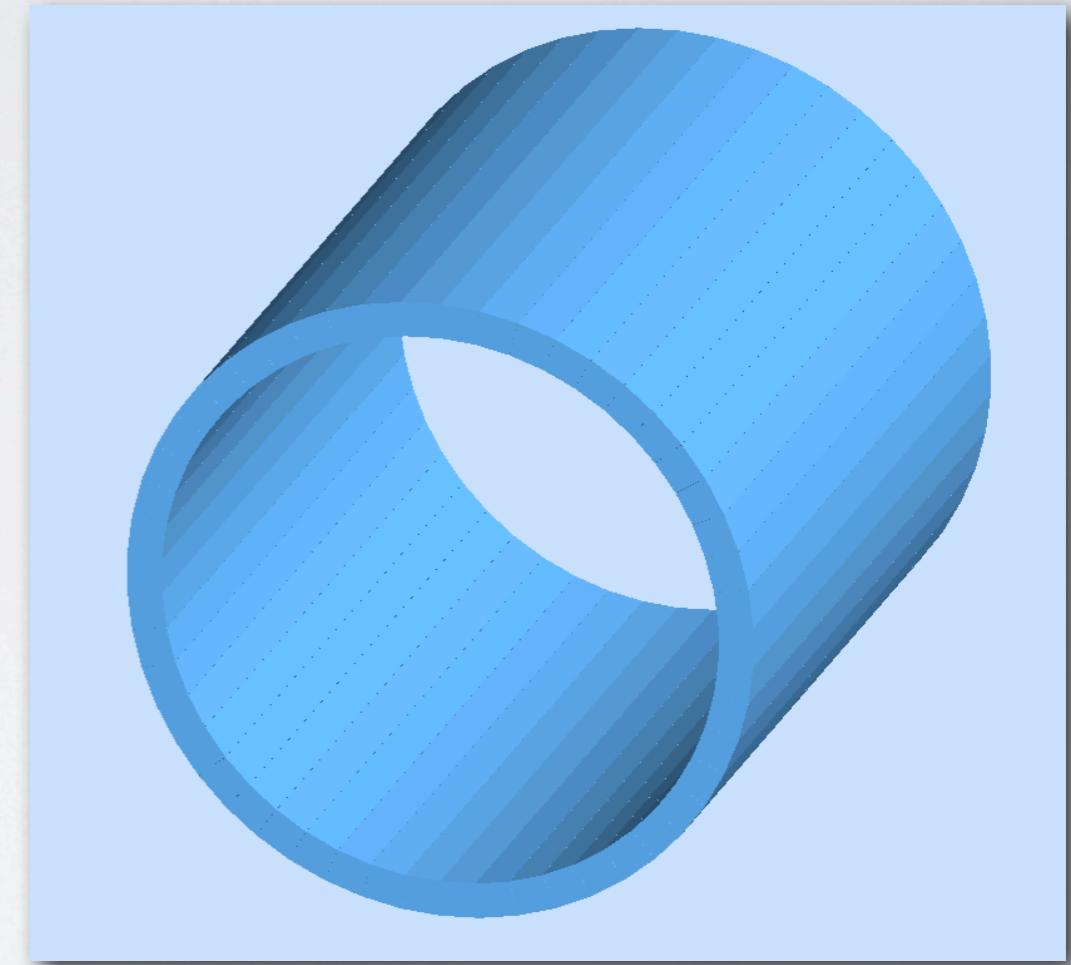
        my %detector = init_det();

        $detector{"name"}      = "CTOF_Paddle_$n";
        $detector{"mother"}    = "CTOF" ;
        $detector{"description"} = "Central TOF Scintillator $n";

        $detector{"pos"}       = "$x*cm $y*cm $z*cm";
        $detector{"rotation"}   = "90*deg $theta2*deg 0*deg";
        $detector{"color"}      = "66bbff";
        $detector{"type"}       = "Trd";
        $detector{"dimensions"} = "$dx1*cm $dx2*cm $dy*cm $dy*cm $dz*cm";
        $detector{"material"}   = "Scintillator";
        $detector{"mfield"}     = "no";
        $detector{"ncopy"}       = $n;

        $detector{"pMany"}      = 1;
        $detector{"exist"}       = 1;
        $detector{"visible"}     = 1;
        $detector{"style"}       = 1;
        $detector{"sensitivity"} = "CTOF";
        $detector{"hit_type"}    = "CTOF";
        $detector{"identifiers"} = "paddle manual $n";

        print_det(\%configuration, \%detector);
    }
}
```



Perl API

GEMC DB I/O

config.dat

```
sub make_TOF
{
    for(my $n=1; $n<=$NUM_BARS; $n++)
    {
        # positioning
        # The angle $theta is defined off the y-axis (going clockwise) so $x and $y are reversed
        my $theta = ($n-1)*$theta0;
        my $theta2 = $theta + 90;
        my $x     = sprintf("%.3f", $R*cos(rad($theta)));
        my $y     = sprintf("%.3f", $R*sin(rad($theta)));
        my $z     = "0";

        my %detector = init_det();

        $detector{"name"}      = "TOF_Paddle_$n";
        $detector{"mother"}    = "TOF" ;
        $detector{"description"} = "Central TOF Scintillator $n";

        $detector{"pos"}       = "$x*cm $y*cm $z*cm";
        $detector{"rotation"}  = "90*deg $theta2*deg 0*deg";
        $detector{"color"}     = "66bbff";
        $detector{"type"}      = "Trd";
        $detector{"dimensions"} = "$dx1*cm $dx2*cm $dy*cm $dy*cm $dz*cm";
        $detector{"material"}  = "Scintillator";
        $detector{"mfield"}    = "no";
        $detector{"ncopy"}     = $n;

        $detector{"pMany"}     = 1;
        $detector{"exist"}      = 1;
        $detector{"visible"}   = 1;
        $detector{"style"}      = 1;
        $detector{"sensitivity"} = "TOF";
        $detector{"hit_type"}   = "TOF";
        $detector{"identifiers"} = "paddle manual $n";

        print_det(\%configuration, \%detector);
    }
}
```

```
# Configuration file for TOF

# Detector name and variation
detector_name: TOF
variation: Original

# Factory can be MYQL or TEXT
factory: MYSQL

# run ranges and variation will apply to geometry,
# materials and parameters
rmin: 1
rmax: 10000

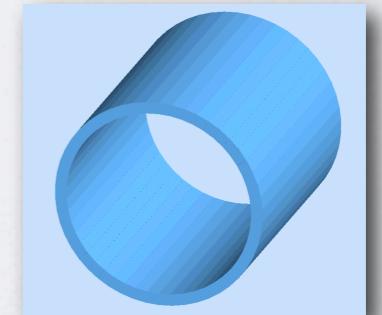
# MYSQL server and database if MYSQL factory is selected
dbhost: clasdb.jlab.org

# Comments/Description
comment: This is the original implementation of the TOF

# Verbosity controls the perl script output
verbosity: 1
```

ctof.pl: Will load the geometry onto CLADB server***

***if you have passport and visa



GEMC DB I/O

config.dat

```
sub make_ITOF
{
    for(my $n=1; $n<=$NUM_BARS; $n++)
    {
        # positioning
        # The angle $theta is defined off the y-axis (going clockwise) so $x and $y are reversed
        my $theta = ($n-1)*$theta0;
        my $theta2 = $theta + 90;
        my $x     = sprintf("%.3f", $R*cos(rad($theta)));
        my $y     = sprintf("%.3f", $R*sin(rad($theta)));
        my $z     = "0";

        my %detector = init_det();

        $detector{"name"}      = "CTOF_Paddle_$n";
        $detector{"mother"}    = "CTOF" ;
        $detector{"description"} = "Central TOF Scintillator $n";

        $detector{"pos"}       = "$x*cm $y*cm $z*cm";
        $detector{"rotation"}  = "90*deg $theta2*deg 0*deg";
        $detector{"color"}     = "66bbff";
        $detector{"type"}      = "Trd";
        $detector{"dimensions"} = "$dx1*cm $dx2*cm $dy*cm $dy*cm $dz*cm";
        $detector{"material"}  = "Scintillator";
        $detector{"mfield"}    = "no";
        $detector{"ncopy"}     = $n;

        $detector{"pMany"}     = 1;
        $detector{"exist"}      = 1;
        $detector{"visible"}   = 1;
        $detector{"style"}      = 1;
        $detector{"sensitivity"} = "CTOF";
        $detector{"hit_type"}   = "CTOF";
        $detector{"identifiers"} = "paddle manual $n";

        print_det(\%configuration, \%detector);
    }
}
```

```
# Configuration file for CTOF

# Detector name and variation
detector_name: CTOF
variation: Original

# Factory can be MYQL or TEXT
factory: MYSQL

# run ranges and variation will apply to geometry,
# materials and parameters
rmin: 1
rmax: 10000

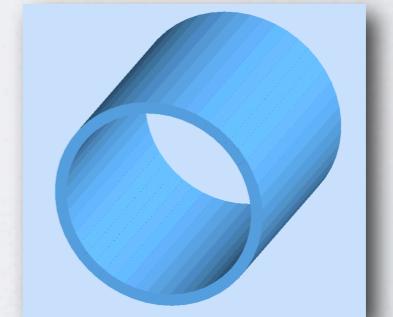
# MySQL server and database if MySQL factory is selected
dbhost: clasdb.jlab.org

# Comments/Description
comment: This is the original implementation of the CTOF

# Verbosity controls the perl script output
verbosity: 1
```

ctof.pl: Will load the geometry onto CLADB server

Geometry immediately available worldwide, w/o recompiling the code



GEMC DB I/O

config.dat

```
sub make_ITOF
{
    for(my $n=1; $n<=$NUM_BARS; $n++)
    {
        # positioning
        # The angle $theta is defined off the y-axis (going clockwise) so $x and $y are reversed
        my $theta = ($n-1)*$theta0;
        my $theta2 = $theta + 94;
        my $x     = sprintf("%.3f", $R*cos(rad($theta)));
        my $y     = sprintf("%.3f", $R*sin(rad($theta)));
        my $z     = "0";

        my %detector = init_det();

        $detector{"name"}      = "CTOF_Paddle_$n";
        $detector{"mother"}    = "CTOF" ;
        $detector{"description"} = "Central TOF Scintillator $n";

        $detector{"pos"}       = "$x*cm $y*cm $z*cm";
        $detector{"rotation"}  = "90*deg $theta2*deg 0*deg";
        $detector{"color"}     = "66bbff";
        $detector{"type"}      = "Trd";
        $detector{"dimensions"} = "$dx1*cm $dx2*cm $dy*cm $dy*cm $dz*cm";
        $detector{"material"}  = "Scintillator";
        $detector{"mfield"}    = "no";
        $detector{"ncopy"}     = $n;

        $detector{"pMany"}     = 1;
        $detector{"exist"}      = 1;
        $detector{"visible"}   = 1;
        $detector{"style"}      = 1;
        $detector{"sensitivity"} = "CTOF";
        $detector{"hit_type"}   = "CTOF";
        $detector{"identifiers"} = "paddle manual $n";

        print_det(\%configuration, \%detector);
    }
}
```

Configuration file for CTOF

Detector name and variation
detector_name: CTOF
variation: Maurizio_1

Factory can be MYQL or TEXT
factory: MYSQL

run ranges and variation will apply to geometry,
materials and parameters
rmin: 22
rmax: 23

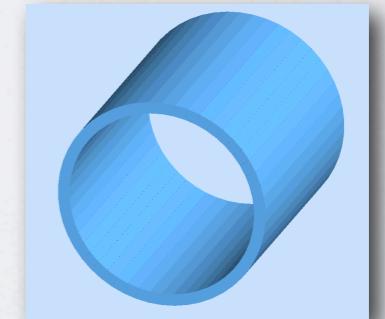
MYSQL server and database if MySQL factory is selected
dbhost: clasdb.jlab.org

Comments/Description
comment: This is the original implementation of the CTOF

Verbosity controls the perl script output
verbosity: 1

ctof.pl: Will load the geometry onto CLADB server

Can choose different variation, run number



GEMC DB Structure

One DB only.

System “DC”

Sub Systems:

Geometry

Materials

Parameters

Bank

Hit

GEMC DB Structure

System “DC”

Sub Systems:

Geometry

Materials

Parameters

Bank

Hit

- ♦ name
- ♦ description
- ♦ tag
- ♦ type
- ♦ rmin
- ♦ rmax
- ♦ variation

GEMC DB Structure

System “DC”

Sub Systems:

Geometry

Materials

Parameters

Bank

Hit

- name
- description
- identifier
- signal threshold
- time window
- production threshold
- max step

GEMC DB Structure

System “DC”

Sub Systems:

Geometry

Materials

Parameters

Bank

Hit

- ❖ name
- ❖ description
- ❖ identifier cell layer superlayer sector
- ❖ signal threshold
- ❖ time window
- ❖ production threshold
- ❖ max step

GEMC DB Security

I Administrator, controls I DB

Developers of a system are given a “passport” and a “visa”.

They are allowed to modify only their system.

Protect systems against mistakes
DB history is saved



```
<detector name="beamline" factory="MYSQL" variation="maurizio_1" r="22"/>
```

GEMC DB Security

I Administrator, controls I DB

Developers of a system are given a “passport” and a “visa”.

They are allowed to modify only their system.

Protect systems against mistakes
DB history is saved



```
<detector name="beamline" factory="MYSQL" variation="maurizio_1" r="22"/>
```

Only the administrator can copy official variations in the “main” part of the DB



```
<detector name="beamline" factory="MYSQL" variation="main: maurizio_1" r="22"/>
```

GEMC GDML I/O

```
<define>
  <rotation name="identity" x="0" y="0" z="0" />
  <position name="shiftbyx" x="-500"/>
  <position name="pos1" x="-250"/>
</define>

<solids>
  <box name="WorldBox" x="wextent" y="wextent" z="wextent"/>
  <cone name="c1" z="111.0" rmax1="22.0" rmax2="33.0" deltaphi="TW0PI" aunit="rad"/>
</solids>

<structure>
  <volume name="vol0">
    <materialref ref="Water"/>
    <solidref ref="b100"/>
  </volume>

  <volume name="vol1">
    <materialref ref="Water"/>
    <solidref ref="c1"/>
  </volume>

  <physvol>
    <volumeref ref="vol0"/>
    <positionref ref="center"/>
    <rotationref ref="identity"/>
  </physvol>

  <physvol>
    <volumeref ref="vol1"/>
    <positionref ref="pos1"/>
    <rotationref ref="identity"/>
  </physvol>
</structure>
```

GEMC GDML I/O

```
<define>
  <rotation name="identity" x="0" y="0" z="0" />
  <position name="shiftbyx" x="-500"/>
  <position name="pos1" x="-250"/>
</define>

<solids>
  <box name="WorldBox" x="wextent" y="wextent" z="wextent"/>
  <cone name="c1" z="111.0" rmax1="22.0" rmax2="33.0" deltaphi="TW0PI" aunit="rad"/>
</solids>

<structure>
  <volume name="vol0">
    <materialref ref="Water"/>
    <solidref ref="b100"/>
  </volume>

  <volume name="vol1">
    <materialref ref="Water"/>
    <solidref ref="c1"/>
  </volume>

  <physvol>
    <volumeref ref="vol0"/>
    <positionref ref="center"/>
    <rotationref ref="identity"/>
  </physvol>

  <physvol>
    <volumeref ref="vol1"/>
    <positionref ref="pos1"/>
    <rotationref ref="identity"/>
  </physvol>
</structure>
```

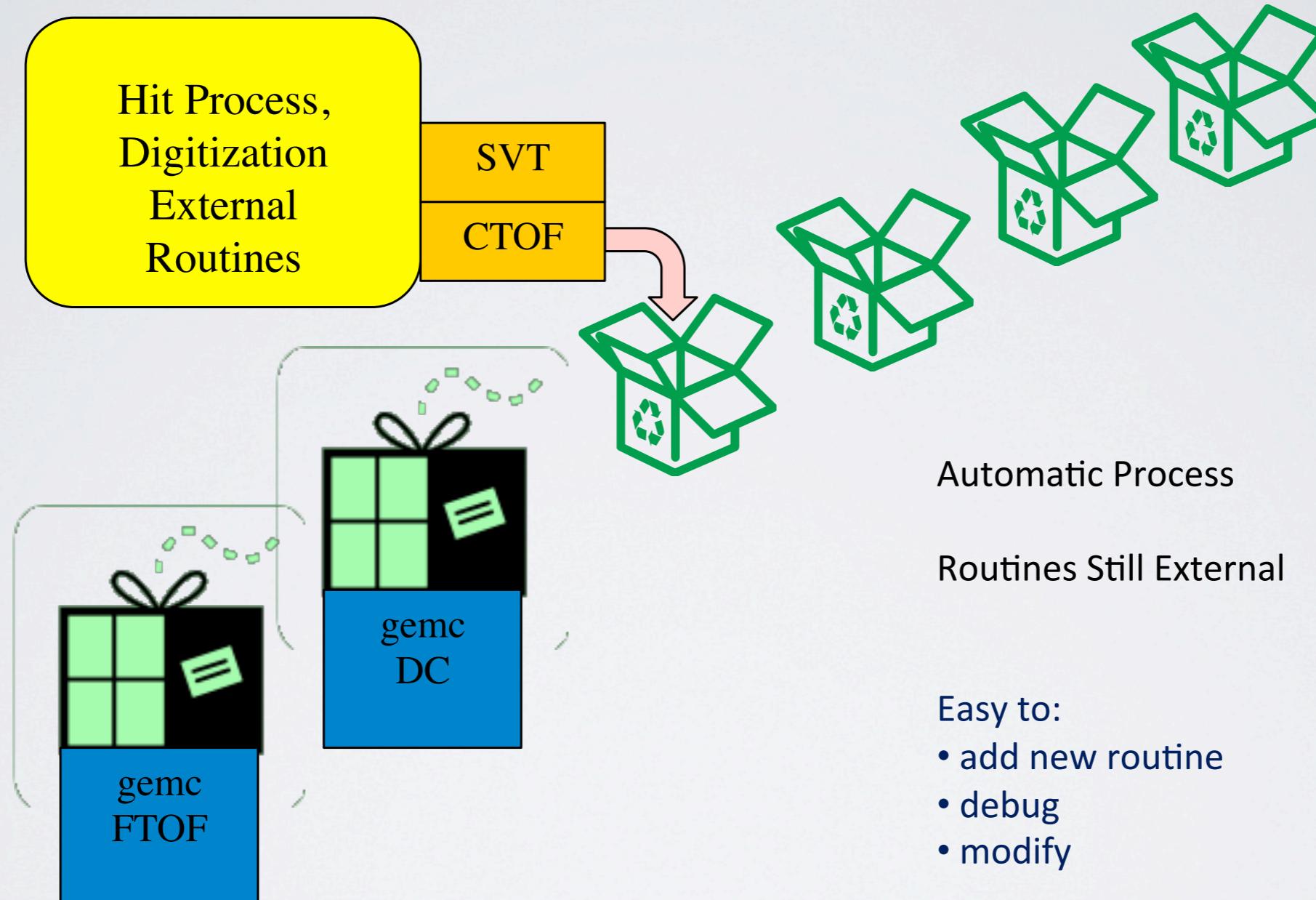
Extensions
HDDS (hall-d)
LCDD (slac lcsim)

GEMC Magnetic Fields

Magnetic Field Factory allows multiple formats
(ASCII, Binary, EVIO)

```
<mfield>
  <description name="hps_dip2" factory="ASCII" comment="First Frascati Magnet - Map shifted to -172.3898 "/>
  <symmetry type="dipole-y" format="map" integration="RungeKutta" minStep="0.1*mm"/>
  <map>
    <coordinate>
      <first name="longitudinal" npoints="53" min="-331.66" max="328.74" units="mm"/>
      <second name="transverse" npoints="5" min="-34" max="34" units="mm"/>
    </coordinate>
    <shift x="0" y="0" z="263.8298" units="cm"/>
    <field unit="T"/>
    <interpolation type="none"/>
  </map>
</mfield>
-331.66      -34   0.06162524223
-331.66      -17   0.06289467961
-331.66       0    0.06393330544
-331.66       17   0.06289467961
-331.66       34   0.06162524223
-318.96      -34   0.07928198576
-318.96      -17   0.08101303875
-318.96       0    0.08222471178
-318.96       17   0.08101303875
-318.96       34   0.07928198576
```

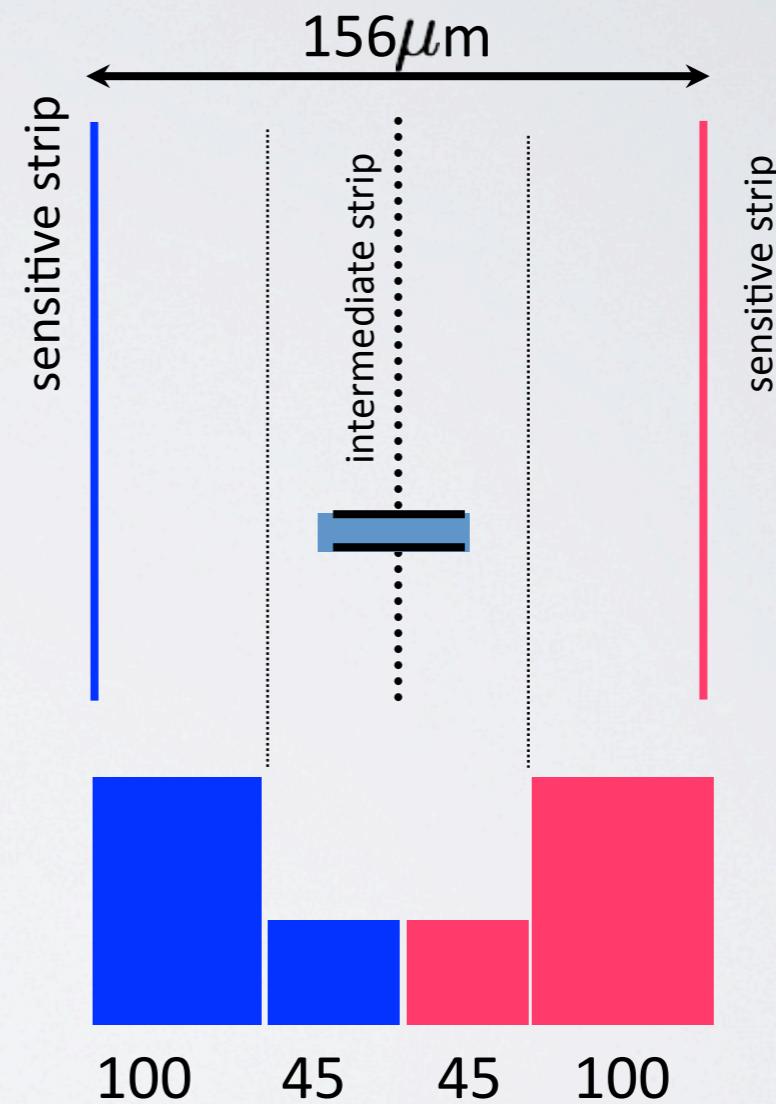
GEMC Digitization



OO approach ensure bugs are virtually non-existent

GEMC Hit Sharing

Edep can be assigned
to (neighbor or not)
detector element



GEMC Hit Process Example

```
PH_output EC_HitProcess :: ProcessHit(MHit* aHit, gemc_opts)
{
    PH_output out;
    out.identity = aHit->GetId();

    // get sector, stack (inner or outer), view (U, V, W), and strip.
    int sector = out.identity[0].id;
    int stack  = out.identity[1].id;
    int view   = out.identity[2].id;
    int strip  = out.identity[3].id;

    int nsteps = aHit->GetPos().size();

    double Etot = 0;
    double Etota = 0;
    double latt = 0;
    vector<G4double> Edep = aHit->GetEdep();

    for(int s=0; s<nsteps; s++)
    {
        if(attlen>0)
        {
            xlocal = Lpos[s].x();
            ylocal = Lpos[s].y();
            if(view==1) latt=xlocal+(pDx2/(2.*pDy1))*(ylocal+pDy1);
            if(view==2) latt=BA*(pDy1-ylocal)/2./pDy1;
            if(view==3) latt=BA*(ylocal+pDy1-xlocal*2*pDy1/pDx2)/4./pDy1;
            Etot = Etot + Edep[s];
            //cout<<"xlocal="<<xlocal<<" ylocal="<<ylocal<<" view="<<view<<" strip="<<strip<<" latt="<<latt<<endl;
            Etota = Etota + Edep[s]*exp(-latt/attlen);
        }
        else
        {
            Etot = Etot + Edep[s];
            Etota = Etota + Edep[s];
        }
    }

    // Jerry Gilfoyle, Feb, 2010
    // parameters: changed from hardwiring (see below) to gpars array - gilfoyle 8/31/12
    //int ec_tdc_time_to_channel = (int) gpars["EC/ec_tdc_time_to_channel"]; // conversion from time (ns) to TDC channels.
    //double ECfactor = (double) gpars["EC/ECfactor"]; // number of photons divided by the energy deposited in MeV; value taken from gsim. see EC NIM paper table 1.
    //int EC_TDC_MAX = (int) gpars["EC/EC_TDC_MAX"]; // max value for EC tdc.
    //int ec_charge_to_channel = (int) gpars["EC/ec_charge_to_channel"]; // conversion from charge (pC) to ADC channels; taken from IC code.
    //double ec_npe_to_charge = (double) gpars["EC/ec_npe_to_charge"]; // unjustified (!! ) conversion from number of photoelectrons to charge in pC.

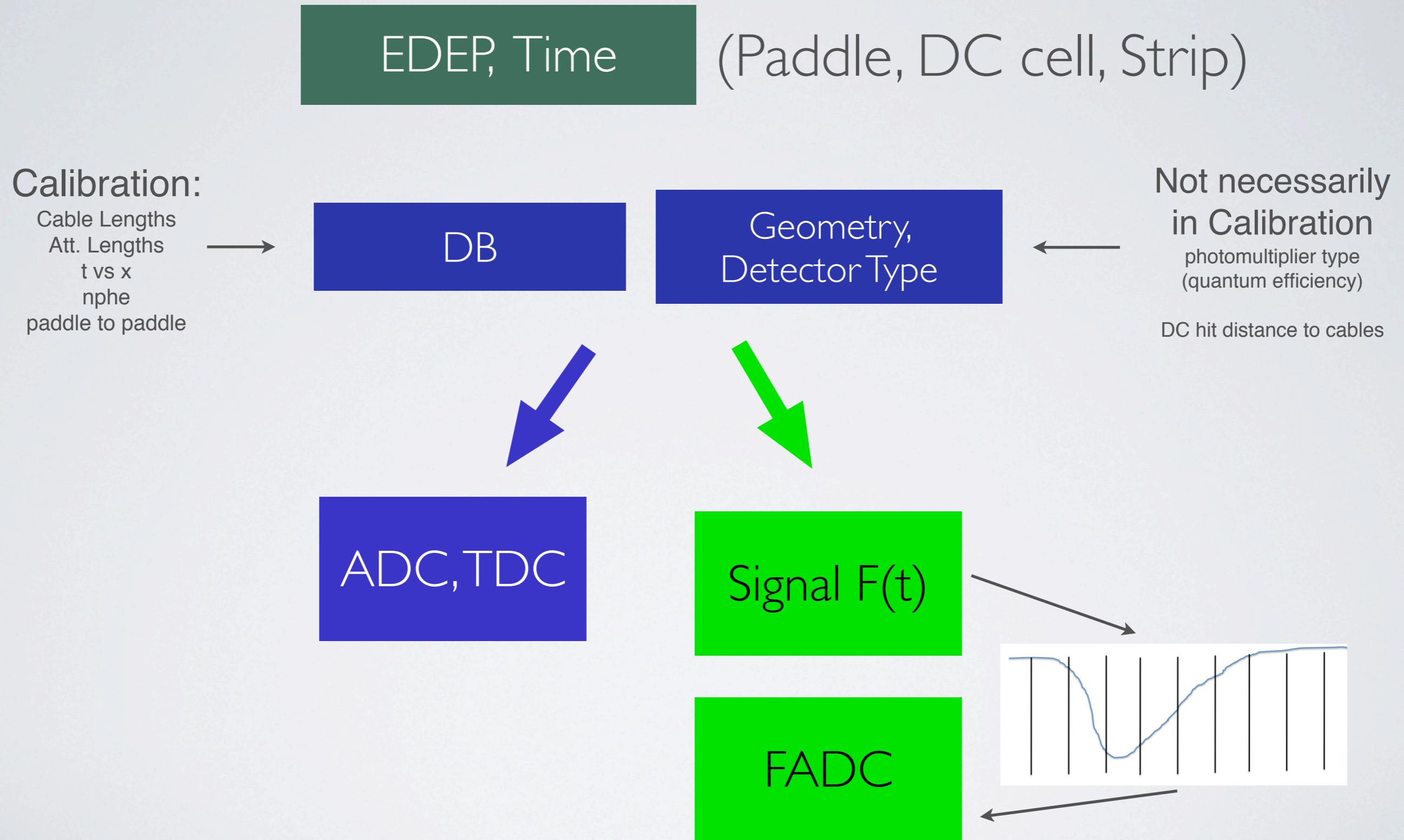
    // parameters were initially hardwired into the code with the values shown below.
    // Moved into trunk/database_io/clas/geo/ec/parameters.txt - gilfoyle 8/31/12

    // simulate the adc value.
    if (Etota > 0) {
        double EC_npe = G4Poisson(Etota*ECfactor); //number of photoelectrons
        // Fluctuations in PMT gain distributed using Gaussian with
        // sigma SNR = sqrt(ngamma)/sqrt(del/del-1) del = dynode gain = 3 (From RCA PMT Handbook) p. 169)
        // algorithm, values, and comment above taken from gsim.
        double sigma = sqrt(EC_npe)/1.22;
        double EC_MeV = G4RandGauss::shoot(EC_npe,sigma)*ec_MeV_to_channel/ECfactor;
        if (EC_MeV <= 0) EC_MeV=0.0; // guard against weird, rare events.
        EC_ADC = (int) EC_MeV;
    }

    // simulate the tdc.
    EC_TDC = (int) (time*ec_tdc_time_to_channel);
    if (EC_TDC > EC_TDC_MAX) EC_TDC = EC_TDC_MAX;

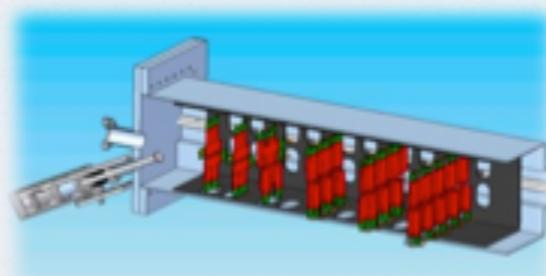
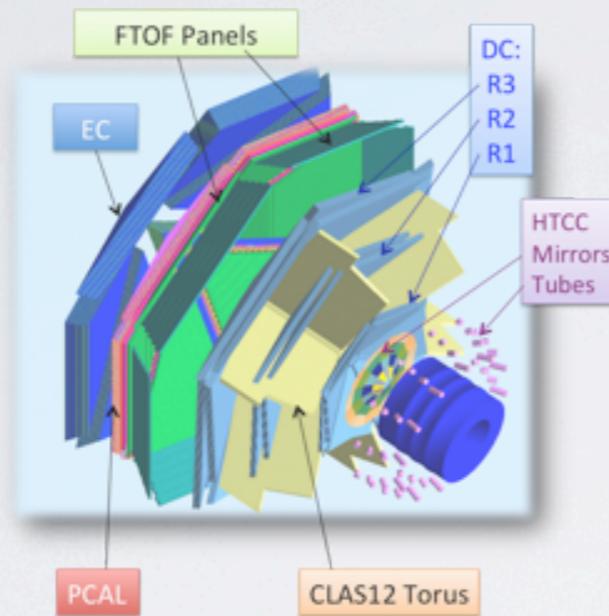
    return out;
}
```

Digitization Flowchart



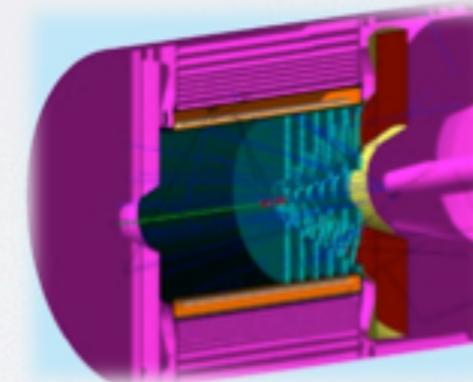
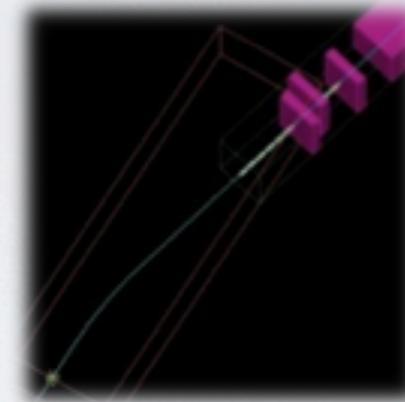
GEMC USE

CLAS12 Detector



HPS Beamlne

GLUEX Beamlne



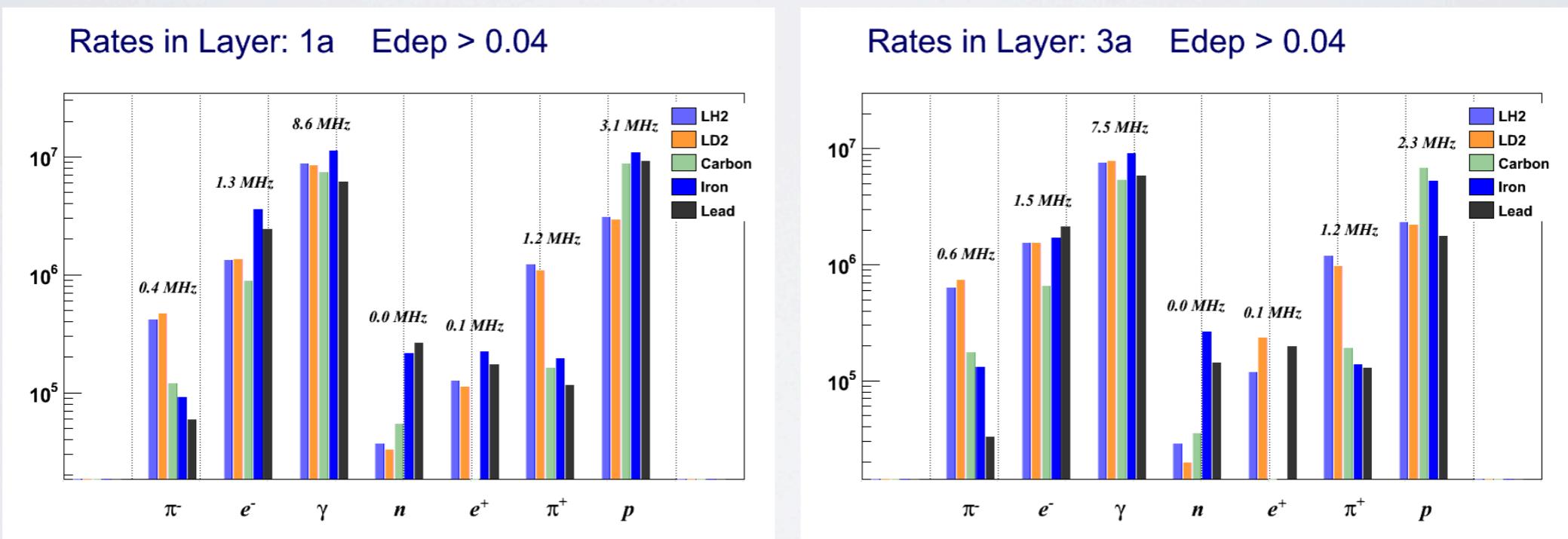
Solid: Hall-A 12

GEMC USE

$E > 0.04 \text{ MeV}$

All targets

Rate written on top is on LH2



GEMC USE

1 sensor = 3.26 g, 46.88cm²

SL1:	3 sensor	10 sectors
SL2:	3 sensors	14 sectors
SL3:	3 sensors	18 sectors

Layer 1a

target	GeV/s	GeV/(s cm ²)	mrad/s	mrad/(scm ²)	rad/year	rad/(year cm ²)
1h2	20325	15.054	6.244	0.00462	196939	145
1d2	20332	15.060	6.247	0.00462	197013	145
C	32220	23.865	9.899	0.00733	312193	231
Fe	52182	38.650	16.032	0.01187	505612	374
Pb	66000	48.885	20.278	0.01501	639498	473

GEMC 3D Output, Web Interface

VRML Ouput

Can be read by free software to
produce 3D PDF

Web Interface in the making
produces images, can run GEMC
on the JLAB farm

Documentation

gemc.jlab.org

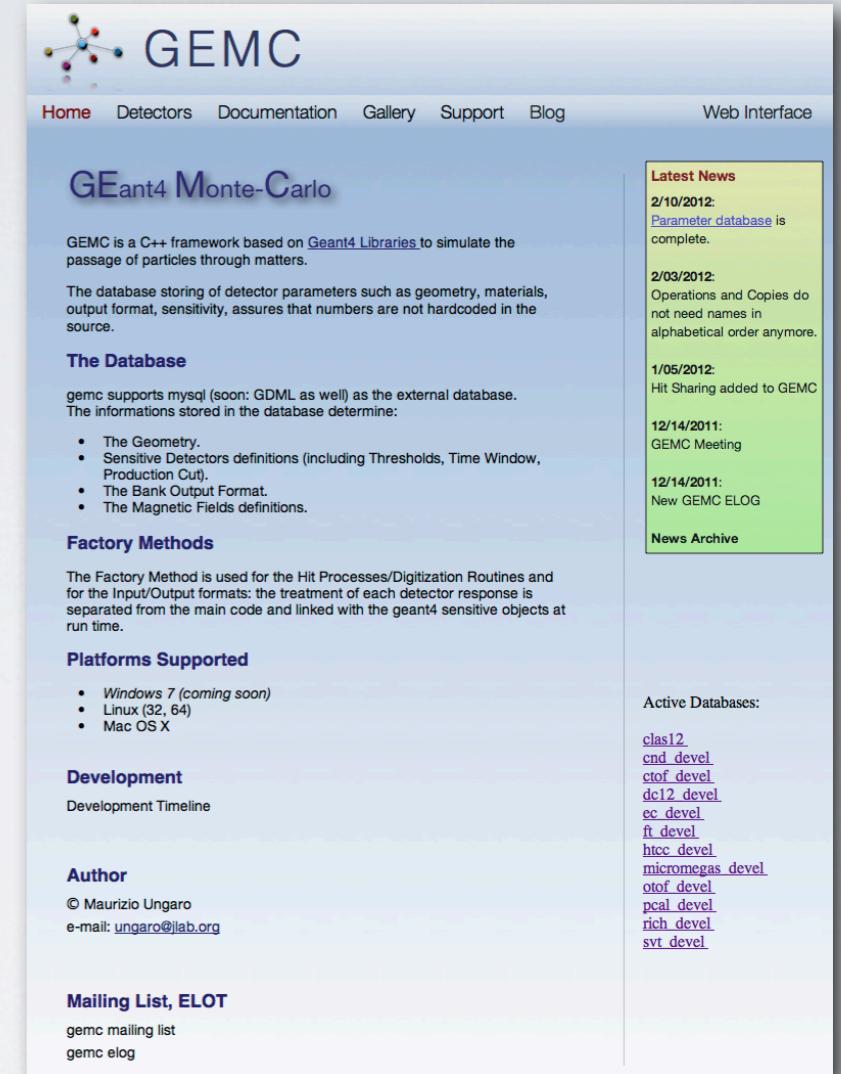
Doxxygen
Installation
HowTos
Detectors

<https://clasweb.jlab.org/elog-gemc/>

Mailing List: gemc_software

Bug Report: Mantis

Doxxygen



GEMC DB I/O

```
sub make_ITOF
{
    for(my $n=1; $n<=$NUM_BARS; $n++)
    {
        my $pnumber      = cnumber($n-1, 10);

        # positioning
        # The angle $theta is defined off the y-axis (going clockwise) so $x and $y are reversed
        my $theta   = ($n-1)*$theta0;
        my $theta2 = $theta + 90;
        my $x      = sprintf("%.3f", $R*cos(rad($theta)));
        my $y      = sprintf("%.3f", $R*sin(rad($theta)));
        my $z      = "0";

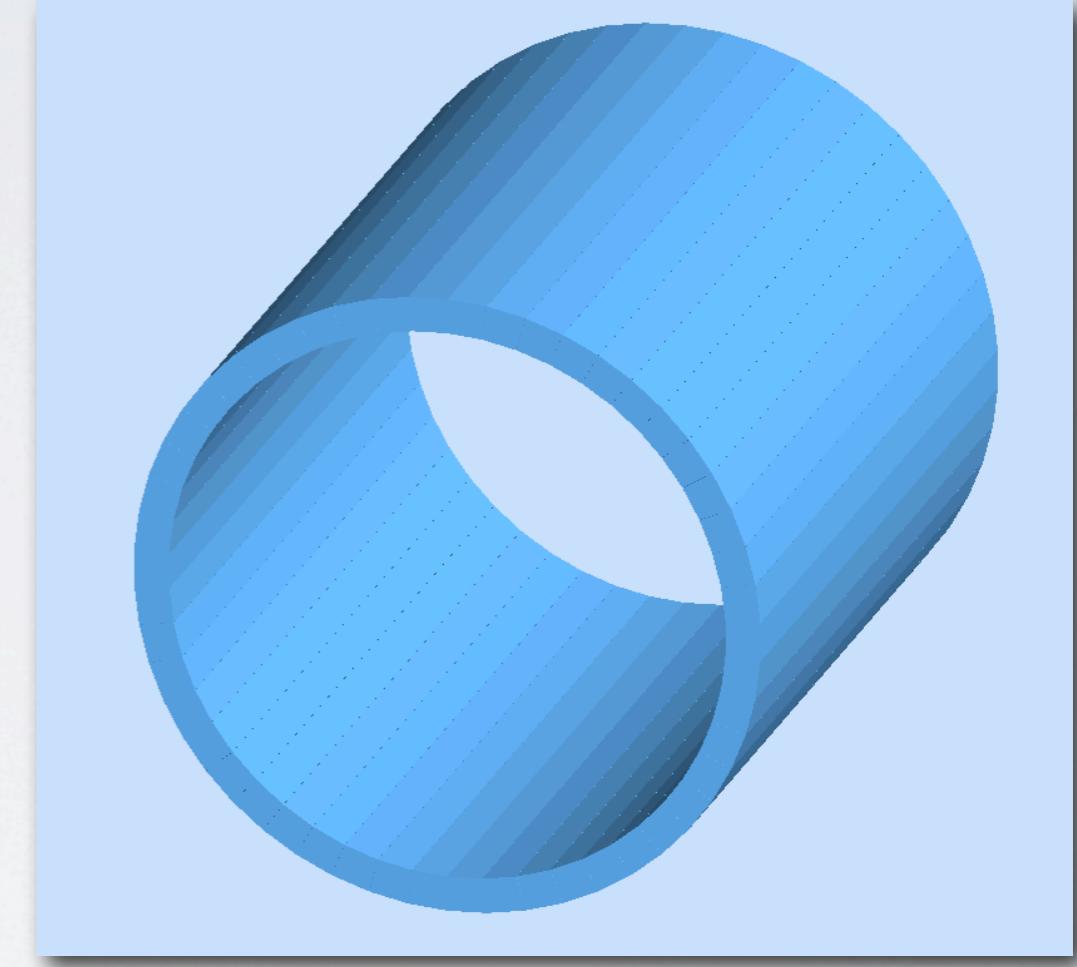
        my %detector = init_det();

        $detector{"name"}      = "CTOF_Paddle_$pnumber";
        $detector{"mother"}     = "CTOF" ;
        $detector{"description"} = "Central TOF Scintillator $n";

        $detector{"pos"}       = "$x*cm $y*cm $z*cm";
        $detector{"rotation"}   = "90*deg $theta2*deg 0*deg";
        $detector{"color"}      = "66bbff";
        $detector{"type"}       = "Trd";
        $detector{"dimensions"} = "$dx1*cm $dx2*cm $dy*cm $dy*cm $dz*cm";
        $detector{"material"}   = "Scintillator";
        $detector{"mfield"}     = "no";
        $detector{"ncopy"}      = $n;

        $detector{"pMany"}      = 1;
        $detector{"exist"}       = 1;
        $detector{"visible"}     = 1;
        $detector{"style"}       = 1;
        $detector{"sensitivity"} = "CTOF";
        $detector{"hit_type"}    = "CTOF";
        $detector{"identifiers"} = "paddle manual $n";

        print_det(\%configuration, \%detector);
    }
}
```



Parameters in DB:
NUM_BARS, Theta0, R, DX1, DX2, DY, DZ