DD4hep: a Detector Description Solution for High Energy Physics Experiments

M.Frank¹, F.Gaede², S.Lu², N.Nikiforou¹, M.Petric¹, A.Sailer¹

¹CERN, ²DESY
Motivation and goals

- **Complete Detector Description**
  - Includes geometry, materials, visualization, readout, alignment, calibration, etc.

- **Support Full Experiment life cycle**
  - Detector concept development, detector optimization, construction, operation
  - Easy transition from one phase to the next

- **Consistent Description, Single source of information**
  - Use in simulation, reconstruction, analysis, etc.

- **Ease of Use**

- **Few places to enter information**

- **Minimal dependencies**
DD4hep Components

- **DD4hep**: basics/core  
  - Basically stable
- **DDG4**: Simulation using Geant4  
  - Validation ongoing
- **DDRec**: Reconstruction support  
  - Driven by LC Community
- **DDAlign, DDCond**: Alignment and Conditions support  
  - Being developed

- [http://aidasoft.web.cern.ch/DD4hep](http://aidasoft.web.cern.ch/DD4hep)
Current Toolkit Users

<table>
<thead>
<tr>
<th></th>
<th>DD4hep</th>
<th>DDG4</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ILD</strong></td>
<td>F. Gaede et al., ported complete model ILD_o1_v05 from previous simulation framework (Mokka)</td>
<td>✓</td>
</tr>
<tr>
<td><strong>CLICdp</strong></td>
<td>New detector model being implemented after CDR, geometry under optimization</td>
<td>✓</td>
</tr>
<tr>
<td><strong>FCC-eh</strong></td>
<td>P. Kostka et al.</td>
<td>✓</td>
</tr>
<tr>
<td><strong>FCC-hh</strong></td>
<td>A. Salzburger et al.</td>
<td>✓</td>
</tr>
</tbody>
</table>

Feedback from users is invaluable and helps shaping DD4hep!
What is Detector Description

- Description of a tree-like hierarchy of "detector elements"
  - Subdetectors or parts of subdetectors

- Detector Element describes
  - Geometry
  - Environmental conditions
  - Properties required to process event data
  - Extensions (optionally): experiment, sub-detector or activity specific data, measurement surfaces, …

M. Frank

17 September 2015
N.Nikiforou, ROOT Users' Workshop 2015  17 September 2015
DD4hep – The Big Picture

(With the CLIC/ILD use case as an example)

Compact description
xml

Detector constructors
C++
python

Generic Detector Description Model
Based on ROOT TGeo
C++

Geometry Display
(Via ROOT OpenGL Viewer)
geoDisplay
tevDisplay

Extensions where required

GDML Converter
xml

TGeo ⇒ G4 converters
Reconstruction Extensions
Analysis Extensions

Alignment / Calibration

Conditions DB

Alignment / Calibration

ROOT GDML Visualization, Geant4, …

Geant4 Program
ddsim

Event Display
(less geometry detail)
CED

Reconstruction Program
Marlin

Analysis Program

M. Frank

N.Nikiforou, ROOT Users' Workshop 2015 17 September 2015
Subdetector Hierarchy (Tree)

- Detectors
  - DetectorElement
    - PlacedVolume
      - [TGeoNode]
      - [TGeoMatrix]

Geometry

GDML content

Subdetector status (conditions)

- Alignment
- Conditions
- Readout
- Visualization
- Segmentation

Material

Envelope

[TGeoShape]

[TGeoBox] [TGeoCone] [TGeoTube]
Detector examples

```
<detector id="DetID_HCAL_Barrel" name="HCalBarrel" type="HCalBarrel_o1_v01"
readout="HCalBarrelHits" vis="HCALVis" >
<dimensions nsides="HCal_symm" rmin="HCal_Rin" z="HCal_Z" />
<layer repeat="(int) HCal_layers" vis="HCalLayerVis" >
    <slice material="Steel235" thickness="0.5*mm" vis="AbsVis"/>
    <slice material="Steel235" thickness="19*mm" vis="AbsVis"/>
    <slice material="Polysterene" thickness="3*mm" sensitive="yes"/>
    <slice material="PCB" thickness="0.7*mm"/>
    <slice material="Air" thickness="2.7*mm"/>
</layer>
</detector>
```

- Fairly scalable and flexible drivers (Generic driver palette available)
- Visualization, Radii, Layer/module composition in compact xml, volume building in C++ driver
- User decides balance between detail and flexibility
- Usually could do a lot just by modifying the xml. For example:
  - Scale detector
  - Create double layers
  - Create "spiral" endcap geometry
  - …
**Good practice**: each subdetector should be contained in an **envelope** defining its boundaries

- Fairly complex envelopes can be fully described in the XML
- Using high-level parameters
  - e.g. Inner/outer radius

```xml
<envelope vis="ILD_ECALVis">
  <shape type="BooleanShape" operation="Subtraction" material="Air">
    <shape type="BooleanShape" operation="Subtraction" material="Air">
      <shape type="BooleanShape" operation="Intersection" material="Air">
        <shape type="Box" dx="R_out" dy="R_out" dz="Z_max"/>
        <shape type="PolyhedraRegular" numsides="symmetry" rmin="0" rmax="R_out" dz="2.0*Z_max"/>
        <rotation x="0*deg" y="0*deg" z="90*deg-180*deg/symmetry"/>
      </shape>
    </shape>
    <shape type="Box" dx="R_in" dy="R_in" dz="Z_max"/>
  </shape>
</envelope>
```

- Envelope placed with a single line in the C++ driver
  ```cpp
  Volume envelope = XML::createPlacedEnvelope(lcdd, element, sdet);
  if (lcdd.buildType()==BUILD_ENVELOPE) return sdet;
  ```

- Use flag in `geoDisplay` to build a simplified geometry using only the envelopes
  - e.g. ILD Detector envelopes
DDRec: Reconstruction extensions

Extend subdetector driver with arbitrary user data

- Summary of more abstract information useful for reconstruction
- Populate during driver construction
  - Driver has the all the information
  - Take advantage of material map
- e.g.: attach a `LayeredCalorimeterStruct` to the `DetElement` for HCalBarrel
  ```cpp
sdet.addExtension<DDRec::LayeredCalorimeterData>(caloData);
```
- Additional simple data structures available
- Users can even attach their own more complicated objects
- Other use cases: auxiliary information for tracking, slimmed-down geometry for a faster event display (e.g. CED[†])

[†] http://ilcsoft.desy.de/portal/software_packages/ced/
Measurement Surfaces

- Special type of extension, used primarily in tracking
  - Did not find an implementation in TGeo
  - Implemented in DDRec
- Attached to **DetElements** and **Volumes** (defining their boundaries)
  - Can be added to drivers via **plugins** without modifying detector constructor
- They hold \( u, v, \text{normal} \) and origin vectors and **inner/outer thicknesses**
- Material properties **averaged automatically**
- Could also be used for **fast simulation**

![Outlines of surfaces drawn in teveDisplay for CLICdp Vertex Barrel and Spiral Endcaps](image-url)
DDG4: Gateway to Geant4

- DD4hep facilitates **in-memory translation of geometry** from TGeo to Geant4
- Plugin Mechanism:
  - Sensitive detectors, **segmentations** and configurable actions, …
- Configuration mechanism (via python, XML, CINT)
  - Physics lists, regions, limits, fields, …
- All shared with **Reconstruction**

**Deposited energy per hit in the CLIC det. HCal**

- Simulated 10 GeV $\mu^-$, uniform in $\phi$
- Fit to a single Landau Distribution
- Fit to a sum of two Landau Distributions

**Second MIP from secondaries (from MC truth history)**

- Detailed validation underway
- Already simulating realistic physics events to develop/test tracking and particle flow-based reconstruction

**Hit map from 100 H$\nu$ events in CLIC det.**
The TGeo Advantage

- Visualize and check the geometry in detail outside Geant4 first with ROOT’s OpenGL viewers
  - Easier manipulation of the scene (rotate, pan, clip, …)
  - Tools (overlap check, independent GDML dump, …)

- Can implement Event Displays using TEve
  - Implement toggling of display of subdetectors on the fly, chose to show just envelopes, just surfaces, …

- Nice treatment of assemblies (especially assemblies-in-assemblies)
Surfaces and Hits in teveDisplay

F. Gaede
Minor inconveniences

- Variety conventions ⇒ can be confusing for the user
- Different conventions between TGeo and Geant4 shape constructors
  - E.g. phi1 and phi2 vs phi1 and dphi
- Different units between TGeo and Geant4
  - Degrees/radians, mm/cm, …
  - Introduced "DD4hep units" (dd4hep::mm, dd4hep::rad, …)
  - We require users to use units explicitly in the compact xml

How standards proliferate: (see: A/C chargers, character encodings, instant messaging, etc.)

Situation: There are 14 competing standards.

14?! Ridiculous! We need to develop one universal standard that covers everyone's use cases. Yeah!

Soon: There are 15 competing standards.

(C) xkcd: 927
Necessary changes implemented, initial tests show that DD4hep compiles and *works* with ROOT 6 and clang/cling.

LC Community are still using ROOT 5: transitioning to ROOT 6 by end of year.

**Major issues encountered:**

- **Abandonment of PluginService** (needed for DD4hep by design)
  - Solution: Use the Gaudi plugin service
- **Problems with OpenGL on ubuntu (?)**
  - Already under investigation by developers
- **TGeo ⇒ VecGeom**: Should not affect us
  - Assuming TGeo interfaces remain the same
Summary and Outlook

- DD4hep provides consistent single source of detector geometry for simulation, reconstruction, analysis
- Takes full advantage of ROOT’s TGeo
- Already in use by LC and FCC Communities
  - Full integration with iLCsoft software framework underway
- Development continues in parallel with validation
- Compatibility with ROOT 6 demonstrated
BACKUP SLIDES
CLIC_SID_CDR Tracker

- Visualized here in geoDisplay
- Around Vertex Detector and beampipe

The same tracker visualized with ROOT's TGeoManager using and intermediate GDML file dumped from Geant4 after loading geometry from DD4hep
LayeredCalorimeterStruct

```
DD4hep::DDRec::LayeredCalorimeterStruct

+ double
+ LayoutType
+ int
+ vector< DD4hep::DDRec::LayeredCalorimeterStruct::Layer >
+ outer_symmetry
+ inner_symmetry
+ layers

+ extent
+ gap0
+ outer_phi0
+ gap1
+ inner_phi0
+ gap2
+ phi0

+ cellSize1
+ absorberThickness
+ inner_thickness
+ outer_nRadiationLengths
+ inner_nRadiationLengths
+ sensitive_thickness
+ distance
+ inner_nInteractionLengths
+ outer_nInteractionLengths
+ thickness
...
```

DD4hep::DDRec::LayeredCalorimeterStruct::Layer

+ outer_thickness
+ cellSize0
Example HCal Barrel Driver

- Always within a function called

```
static Ref_t
create_detector(LCDD& lcdd, xml_he, SensitiveDetector sens)
{
    ...
    return sdet;
}
```

- Macro to declare detector constructor at the end:

```
DECLARE_DETELEMENT(HCalBarrel_v01, create_detector)
```