DD4hep: a Detector Description Solution for High Energy Physics Experiments

Nikiforou, N. (CERN) et al

17 September 2015

The AIDA-2020 Advanced European Infrastructures for Detectors at Accelerators project has received funding from the European Union’s Horizon 2020 Research and Innovation programme under Grant Agreement no. 654168.

This work is part of AIDA-2020 Work Package 3: Advanced software.

The electronic version of this AIDA-2020 Publication is available via the AIDA-2020 web site <http://aida2020.web.cern.ch> or on the CERN Document Server at the following URL: <http://cds.cern.ch/search?p=AIDA-2020-SLIDE-2016-001>

Copyright © CERN for the benefit of the AIDA-2020 Consortium
DD4hep: a Detector Description Solution for High Energy Physics Experiments

M.Frank\textsuperscript{1}, F.Gaede\textsuperscript{2}, S.Lu\textsuperscript{2}, N.Nikiforou\textsuperscript{1}, M.Petric\textsuperscript{1}, A.Sailer\textsuperscript{1}

\textsuperscript{1}CERN, \textsuperscript{2}DESY
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, …
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

Conditions DB

Alignment / Calibration

ROOT GDML Visualization, Geant4, …

Geant4 Program
ddsim

Event Display
(less geometry detail)
CED

Reconstruction Program
Marlin

Analysis Program

M. Frank
Subdetector Hierarchy (Tree)

- Detectors
  - DetectorElement
    - [TGeoShape]
    - [TGeoMatrix]
    - PlacedVolume [TGeoNode]
      - [TGeoBox]
      - [TGeoCone]
      - [TGeoTube]

Geometry

- Alignment
- Conditions
- Readout
- Visualization
- Segmentation
- Material

GDML content

Subdetector status (conditions)

N.Nikiforou, ROOT Users' Workshop 2015  17 September 2015
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
  - …
Envelopes

- **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

- 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
  - `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/](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 \( u, v, \) 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
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
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

![XKCD comic on standards](https://xkcd.com/927/)
DD4hep, ROOT 6 and the Future

- 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 $\Rightarrow$ 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

```xml
<detector name="SiTrackerBarrel" type="SiTrackerBarrel" readout="SiTrackerBarrelHits" reflect="true"/>
```

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

double
LayoutType
int
vector< DD4hep::DDRec::LayeredCalorimeterStruct::Layer >

double
+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

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

- Macro to declare detector constructor at the end:

```cpp
DECLARE_DETELEMENT(HCalBarrel_o1_v01, create_detector)
```