HEP detector description supporting the full experiment life cycle

M. Frank, F. Gaede, M. Petric, A. Sailer
Motivation and Goal

- Develop a detector description
  - For the full experiment life cycle
    - detector concept development, optimization
    - detector construction and operation
    - “Anticipate the unforeseen”
  - Consistent description, single source, supporting
    - simulation, reconstruction, analysis
  - Full description, including
    - Geometry, readout, alignment, calibration etc.
Philosophy of DD4hep & Co

- Effort of very few people with a simple, humble and comprehensive vision

Detector description for the lazy
Minimal effort, pragmatic, no technical restrictions,
No obstacles induced by religious wars

- DD4hep is the “glue”
  - Bring together what belongs together:
    Detector structure, geometry, simulation, conditions, etc
  - Reuse existing modules: TGeo, Geant4, GitCondDB, etc

- ‘Responsible’ users highly welcome
- Contributions even more!
What is Detector Description?

- **Tree-like hierarchy of “detector elements”**
  - Macroscopic (ie. not a strip)
  - Subdetectors or parts of subdetectors

- **Detector Element**
  - Geometry
  - Properties to process events
    - Environmental data
    - Alignments
    - Derivatives of these
  - Optionally experiment, sub-detector or activity specific data
Detector Description is not Geometry (Please update your glossary)

- Detector element tree
- Geometry (pseudo-)tree

- Tracker
- Placement
  - Tracker
    - Placement
      - Tracker Region
        - Placement
          - Tracker Sector
            - Placement
              - Disk
                - Sector A
    - Placement
      - Barrel
        - Sector I
Class Diagram: Detector Element
Sort of Standard...

Subdetector Hierarchy (Tree)

Detector

DetectorElement

PlacedVolume

[TGeoNode]

[TFGeoMatrix]

LogicalVolume

Envelope

[TGeoShape]

[TFGeoBox]  [TFGeoCone]  ...  [TFGeoTube]

Geometry

Alignment

Conditions

Readout

Visualization

Segmentation

0...n

children 1..n
detector: 1
placements: 0...1

transform: 1

0...1

visattr: 0...1
DD4Hep - The Big Picture

Generic Detector Description Model
based on ROOT TGeo

Provided extensions
- GDML Converter
- TGeo → G4 Converter
- Reco. Extensions
- Analysis Extensions

Event Display
- Alignment Calibration

Compact description
- CAD Drawing
- CAD Converter

Detector constructor
- Compact description
- CAD Drawing
- CAD Converter

Detector constructor
- Compact description
- CAD Drawing
- CAD Converter

Detector constructor
- Compact description
- CAD Drawing
- CAD Converter

Generic description
- Conditions DB
- DDDDB Converter
DD4Hep - The Big Picture

Central Description

Provided extensions

Independent Components
PR: CMS Trackers
PR: FCC Design Study
PR: ILD Model ILD_o1_v05
(F.Gaede, L.Shaojun)
Progress on detector simulation

- STCF software team has been formed.
- OSCAR: Offline Software of Super Tau-Charm Facility.
- Detector geometry with DD4hep.
**DD4hep Core**

- Handles the detector element functionality
- Basically stable
  - Bug fixes, enhancements
- Objects and sub-packages are reflective
  - ROOT C++ dictionary defined
  - Intrinsic support for cross-language development
- Reflection supports interactivity
  - Cint (Cling/AClick) and python (cppyy)
- CHEP 2013

*DD4hep: A Detector Description Toolkit for High Energy Physics Experiments*
Views & Extensions: Users Customize Functionality

DD4hep is based on handles (smart pointers)

- Rarely deal with data directly
- Possibility of many views based on the same DE data
  - Same ‘data’ associated to different ‘behaviors’
  - All views are consistent and creation is efficient: pointer-copy
- Be prudent: a blessing and a curse
  - User data are common knowledge!
Simulation: DDG4

- **Simulation** = Geometry + Detector response + Physics

- **Mature status**
  - Eventual bug fixes, smaller improvements
  - Phase of constant re-validation

- **Automatic geometry conversion**

- **Palette of standard sensitive detectors**

- **Support for MC truth handling**

- **CHEP2015**

DDG4 A Simulation Framework based on the DD4hep Detector Description Toolkit
Geant4 Provided Hooks
[and what we want to do inside]

Main issue: flexible configuration

Flexible definition of the physics list
- Define particles, processes, physics constructors or use/extend predefined physics lists

Flexible data input
- Programmable sequence. Input from particle gun, lcio, stdhep or HepMC (text) – easily extensible
- Modules to smear and boost primary vertices
- Modules to construct interaction overlays
- Further extensions may independently added

Provide user programmable sequences
- Either as explicit object type using ABC
- Or registering a member function as callback
Example of an Action Sequence
Event Overlay with Features

- Combine simple and reusable modules
- Input module
  - Any data format
- Primary vertex smearing
- Primary vertex boost
- Common: initialization, final merge
Another Example: MC Truth Handling (LC specific algorithm)

Registers itself as global MC truth handler

- Callback when hit is created
- Connect to stepping action by callback: Remember if track created secondaries
- Connect to begin/end event by callback: Store user track at end
- Connect to begin/end event by callback: Init event related data

Geant4Sensitive

Geant4GeneratorActionSequence

Geant4GeneratorAction

Geant4ParticleHandler

Geant4SteppingActionSequence

Geant4TrackingActionSequence

Geant4EventActionSequence

Automatically called as part of the event generation
External Framework Support

• 2 possibilities
  – DDG4 (Geant4) takes over event loop from framework
  – Framework steers event loop (overloading run manager)

• Everything is a plugin (or could be made one)
  – External frameworks can overload all central entities
    – G4RunManager
    – Geant4 action routines
    – Physics …

• External framework context
  – Typed pointer available to every user action
  – User defined structure allows to interact with any framework service: histograms, I/O, etc.
Fast Simulation & SLIC

- **Fast Simulation:** Support for fast simulation provided by Geant4
  - SFT fellow working on fast simulation in Geant4
  - Some commitment to also handle DD4hep integration
  - Personal guess: Once it is understood what to do in Geant4
    Provide plugins and attach them to sensitive volumes

- **SLIC**
  - Abandoned (though initially written for SiD/LC)
  - Uses old GDML implementation with limited geometry support
  - So much to be said for “standard”

"EIC Geometry Description and detector interface", page 2
DDG4 in Production

- Deployed for CLICdp in DIRAC
  - For every detector study (now ~14) central generation
- ILC started mass production

\[ e^+ e^- \rightarrow H(\rightarrow b\bar{b})\nu\nu \]
Standard Detector Palette
DDDetectors

- Used for design studies (LC, FCC-eh)
- Origin from the SiD detector model
  - Layer based detectors
  - Tracker barrel & endcap
  - Several calorimeter constructs
- Partially with measurement surfaces (F. Gaede)
  - Uses plugin mechanism to enhance detector elements
    - Non intrusive mechanism to attach user defined optional data
      => 'anticipate the unforeseen'
- Sensitive volumes identified by CellID: up to pixel
  - Volume path reduced to 64 bit number
  - CellID in simulated hit + detector description → placement
DDCond: Conditions Data

• Time dependent data necessary to process the detector response [of particle collisions]
  – slowly changing: every run \( O(1h) \), lumi section \( O(10\text{min}) \) ...
  – multiple conditions change in batches: require discipline
  – conditions may be the result of computation(s)

• DDCond deals with the management of these data
  – Efficient and fast, if used according to design ideas
  – Manages resources
  – Supports multi threading by design
    Well defined locking points
  – Cache where necessary but no more

• CHEP2018
  Conditions and Alignment extensions to the DD4hep Detector Description Toolkit
Global and Local Alignments

- **Global alignment corrections**
  - Physically alters geometry
    Intrinsically supported by ROOT
  - By construction not multi-threaded
  - Possibility to simulate misaligned geometries

- **Local alignment corrections**
  - Geometry stays intact (either ideal or globally aligned)
  - Multi-threading supported, multiple versions
  - Local alignment corrections are conditions
  - Provide matrices from ideal geometry to world e.g. to adjust hit positions

- Both supported
Local Alignment \( \Delta \)-Parameters

- Trickle-up the hierarchy and compute the matrices the most effective way with re-use of intermediate results
- Math verified by AIDA\(^{2020}\) alignment task force (C.Burr)
Toolkit Users

Increasing interest in the HEP community

- ILC  F. Gaede et al.
- CLICdp A. Sailer et al.
- SiD   D. Protopopescu et al.
- FCC-eh P. Kostka et al.
- FCC-hh A. Salzburger et al.
- FCC-ee O. Viazlo (CLD design), N. Alipour, G. Voutsinas
- SCTF  Super-Charm-Tau Factory designs (Novosibirsk, Beijing)
- EIC   Evaluation considered/started (W. Armstrong et al.)

- LHCb  LHCb Upgrade for Run III (B. Couturier et al.)
- CMS   Evaluation for upgrade started (202x) (Y. Osborne et al.)
- CALICE Calorimeter R&D, started
Summary

- DD4hep is getting mature
- Flexible components interacting with user framework
- Starts being capable of handling all aspects of detector description for the lifetime of an experiment
- Increasing interest in the community and increasing number of users
- Visit us on:
  - http://dd4hep.cern.ch
  - Up to date doxygen information
  - User Manuals: have improved but not perfect
Questions and Answers
Saga in 5 Episodes

- **DD4hep** – basics/core (1)
- **DDG4** – Simulation using Geant4 (1)
  - Fast simulation (4)
- **DDRec** – Reconstruction supp. (2)
- **DDCond** – Detector conditions (3)
- **DDAlign** – Alignment support (3)
- **DDDigi** – Generic Digitization (4)

(1) Mature state: bug-fixes and maintenance
(2) F. Gaede (WP3, Task 3.6)
(3) Work since start of AIDA 2020
(4) Planned extensions
Multiple Input Sources

(Demonstrated)

Compact description xml

Detector constructors c++

(Demonstrated)

Conditions DB (LHCb)

DDDB converter c++

(Demonstrated)

CMS XML geometry

DDCMS converter c++

(Future)

CAD drawing

CAD converter c++

Generic Detector Description Model

Based on ROOT TGeo c++
Conditions implementation in LHCb Velo Detector

- People want to see “Detector elements”
  - Fully functional description of parts of the detector
    - Long term valid stuff (structure)
    - Short lived quantities (temperature, alignment, …)
- A “natural” aggregation would be similar to:

  ![Diagram](image)

  - Intuitive, but not good: violates multi-threading
Conditions implementation in LHCb Velo Detector

- **Chosen solution:**
  - Use IOV dependent projection for event processing
    - This is our new “detector element”
    - Keeps reference to the not changing properties
  - Dress with facade to provide required functionality(ies)

```
geometry #1  structure #1

ParameterMap  DD__Static

Velo__DE__Static

AlignmentInfo  CalibrationInfo  ....  ReadoutInfo

DE__IOV

d__static #0...1

conditions #0...*

DE__Conditions

Velo__DE__IOV

IOV: [0, infinity]

IOV: [t1, t2]
```