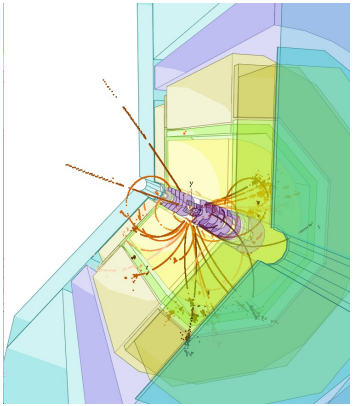
ILC Software

F.Gaede, DESY

Belle II SWWS, DESY, May 14-18, 2018

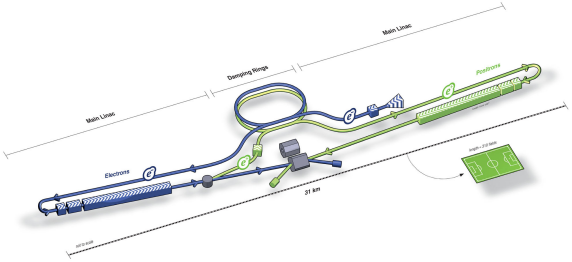
Outline

- Brief overview ILC
- Overview iLCSoft
  - LCIO, DD4hep, Marlin
- Tracking tools in Marlin
- Tracking geometry in DD4hep
- Possible connections to Belle II
- Summary



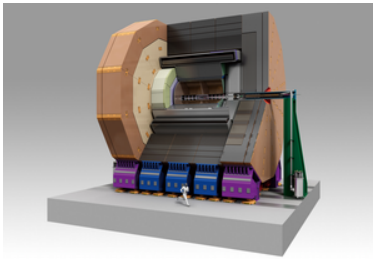
Brief Overview ILC

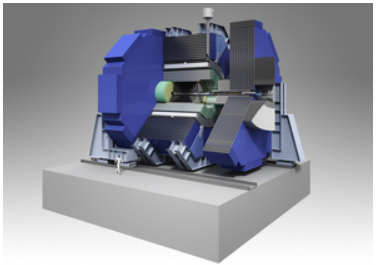
- e^+e^- collider with SCRF
  - $E_{cms} = 250 - 500\ GeV$ (1 TeV)
  - length = 31 km (~50 km)
  - luminosity: $2x10^{34} cm^{-2} s^{-1}$
  - to be build in Japan
- currently the ILC community is looking into *energy staging* scenarios for cost reduction - starting with **250 GeV**
- expect *positive* signals from Japan within this year
  - in time for the *European Strategy Update*



Detectors at the ILC

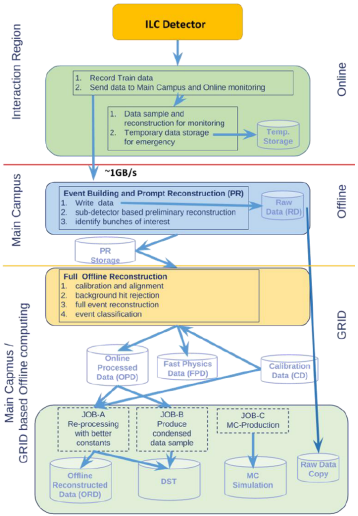
- two detectors in *push-pull* operation
  - optimized for PFA
  - highly granular calorimeters
  - excellent tracking and vertexing
- **ILD**
  - large gaseous tracking: *TPC*
  - size: $\approx 16mx14m$
  - B-field: $3.5Tesla$
- **SiD**
  - all silicon tracker
  - size: $\approx 14mx11m$
  - B-field: $5Tesla$





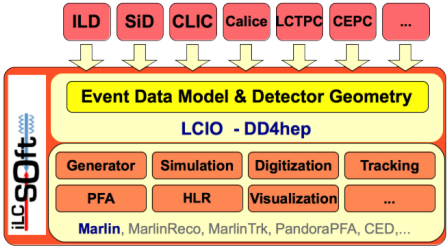
ILC Computing Scheme

- ILC will run without trigger
- expected data rate: **1 GB/s**
  - (at design luminosity)
- total raw data ~**10 PB/a**
- currently we envision a *traditional* computing scheme for the ILC
  - prompt reconstruction and event building *online*
  - full reconstruction and data reduction *offline*
- ILC computing requirements (in > 10 years) will be comparable to those of ATLAS/CMS in RUN 2



Overview iLCSoft

- iLCSoft is the common software framework for Linear Collider detector studies
  - used by ILD, SiD, CLICdp, Calice, LCTPC (and friends: FCC, CEPC, HPS, EIC, ...)
  - and some early conceptual studies for **Belle II**
- key components:
- **LCIO** the common *event data model (EDM)*
- **Marlin** the *application framework*
- **DD4hep** the common *detector geometry description*
- see: `https://github.com/iLCSoft`

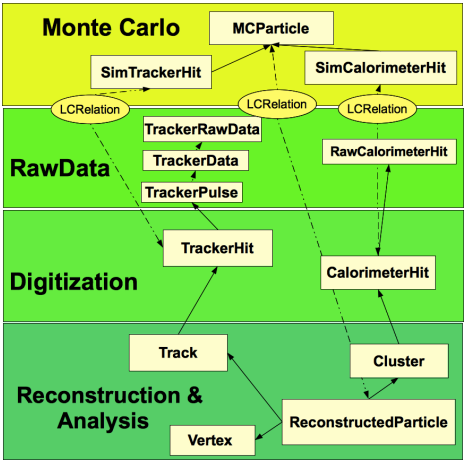


design goals

- modular and flexible
- use and develop *generic HEP tools*, wherever possible

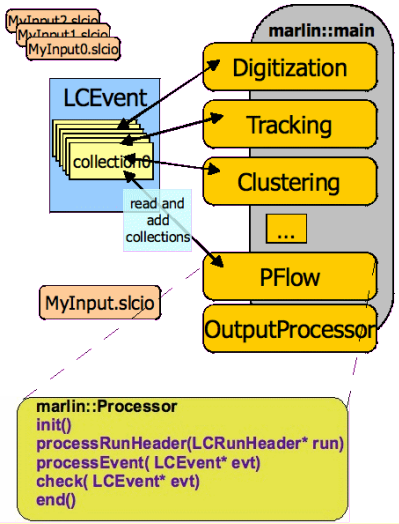
LCIO

- LCIO provides the common *EDM* and *persistency*
- the EDM is hierarchical:
  - you can always get the constituent entities from a higher level object, e.g. the *TrackerHits* that were used to form the *Track*
  - only exception: *you cannot directly go back to the Monte Carlo Truth information*
  - this is possible via dedicated *LCRelation* collections
- everything is stored in *LCCollections*
- collections are retrieved from the *LCEvent* via their **name**
- see: `http://lcio.desy.de`



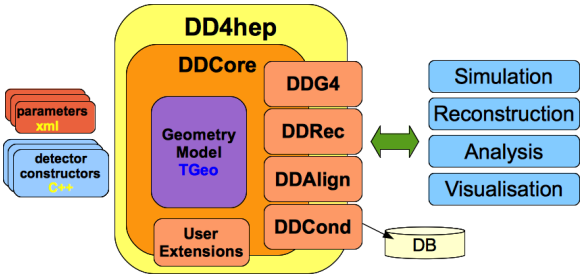
Marlin

- application framework used throughout iLCSoft
- every task is implemented in a *Processor*
  - task can be as trivial as digitizing a hit collection or as complex as running the full *PFA*
- Marlin applications are fully configured via XML files, defining:
  - global parameters
  - the chain of processors to run
  - per processor parameters
- xml files created with *editor* or via *MarlinGUI*
- more: `http://ilcsoft.desy.de/Marlin/current/doc/html/index.html`

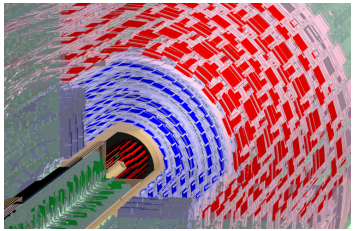


DD4hep - Detector Description Toolkit for HEP

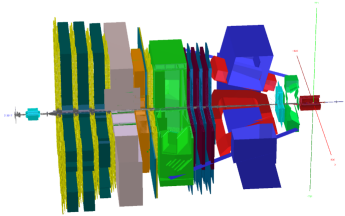
- generic detector description toolkit for any particle physics experiment
- feeds into Simulation, Reconstruction, Analysis, Visualization,...
- component based architecture with interfaces to alignment and conditions data
- uses **ROOT-TGeo** and **Geant4**
- **HSF** and **AIDA2020** project
- currently used by: ILC, CLICdp, CEPC, FCC, interest from LHCb and CMS



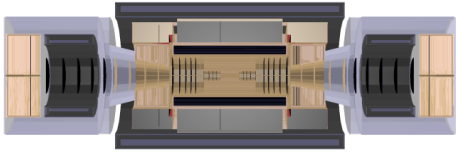
DD4hep non-LC example detectors



CMS



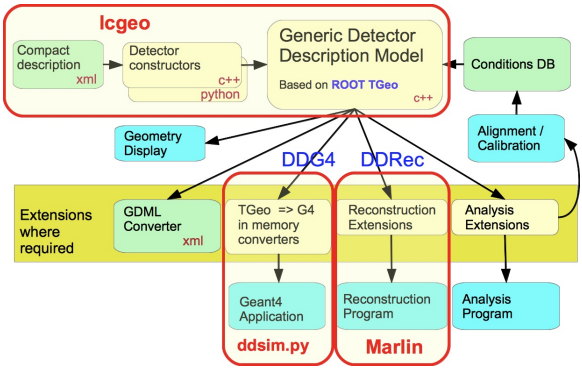
LHCb



FCC-hh

DD4hep for ILC

- DD4hep is used as the detector geometry description in iLCSoft
- the detector is fully described via a set of:
  - C++ detector constructors
  - XML files (*compact files*)
- main components used for ILC:
  - **DDG4** full simulation with Geant4
  - **DDRec** interface for reconstruction (tracking)
- **lcgeo**: sub-package with LC detector models
- **ddsim**: python program to run a full simulation



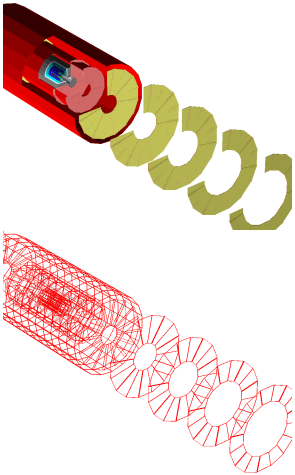
`http://aidasoft.web.cern.ch/DD4hep`

DDRec: dedicated tracking surfaces

- tracking needs special interface to geometry
- measurement and dead material surfaces (planar, cylindrical, conical)
- surfaces attached to volumes in detailed geometry model

surfaces:

- u,v, origin and normal
- inner and outer thicknesses and material properties
- local to global and global to local coordinate transforms:
  - $(x, y, z) \leftrightarrow (u, v)$



DDRec surfaces and materials

- material properties are **automatically averaged**
  - from detailed model
  - along normal of the surface along given thicknesses

averaging materials

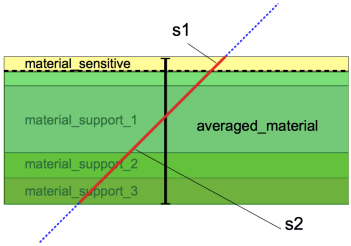
$$< A >= \left(\sum_i^N \rho_i t_i\right) / \left(\sum_i^N \rho_i \frac{t_i}{A_i}\right) \quad t_i \text{ thickness}$$

$$< Z >= \left(\sum_i^N \rho_i \frac{t_i Z_i}{A_i}\right) / \left(\sum_i^N \rho_i \frac{t_i}{A_i}\right) \quad \rho_i \text{ density}$$

$$< \rho >= \left(\sum_i^N \rho_i t_i\right) / \left(\sum_i^N t_i\right)$$

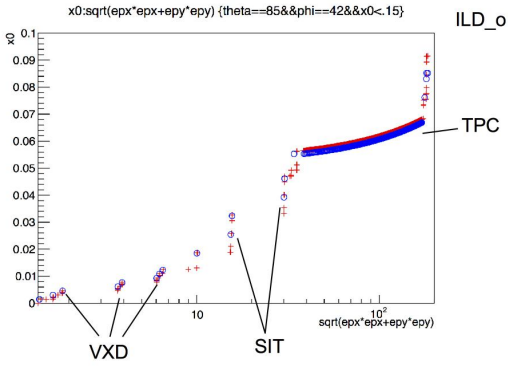
$$< X_0 >= \left(\sum_i^N t_i\right) / \left(\sum_i^N \frac{t_i}{X_{0i}}\right)$$

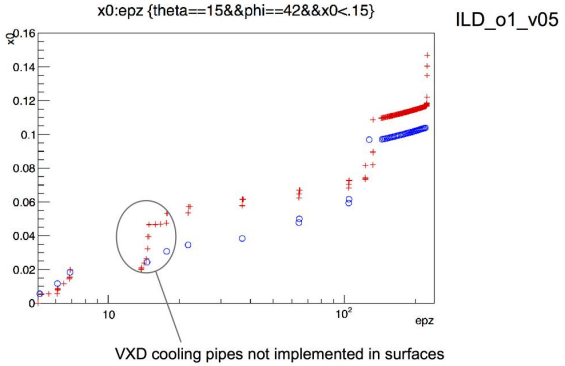
$$< \lambda >= \left(\sum_i^N t_i\right) / \left(\sum_i^N \frac{t_i}{\lambda}\right)$$



roughly equivalent for Bethe-Bloch - identical for multiple scattering

example for ILD: comparison surfaces vs. detailed model

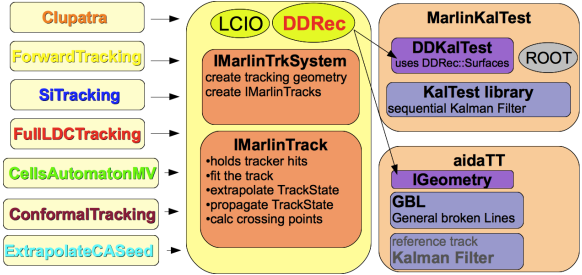




- surface describe material well in most regions - but not everywhere ...
  - would need to add more surfaces in some regions to be *perfect*

LC Tracking tools

- *MarlinTrk* is a generic tracking toolkit
  - based on the **LCIO-EDM** and **DDRec** geometry description
  - used by ILC, CLICdp and CEPC
- *IMarlinTrk* interface separates pattern recognition from concrete track fitters
- have various *stand-alone* and rather *detector independent* pattern recognition algorithms



- plan to evolve into a truly generic HEP tracking toolkit
  - investigate (additional) use of **ACTS** as fitter

Navigation in the tracking geometry

- aidaTT:
  - compute all intersections of reference track with the surfaces and sort wrt. s (path length)
- KalTest/DDKalTest:
  - compute a *global ordering* of all surfaces in the geometry wrt. *one single ordering parameter*:
    - radius of planes and cylinders parallel to z
    - outermost radius of disk like measurement surfaces
    - potentially a source of errors and *not fast*
- ideally we would like to have a smart topological pre-ordering of the surfaces (a la ATLAS)
  - not *strictly needed* but would be *nice to have* for large MC production
  - would like to see what *ACTS* can do for us here ...

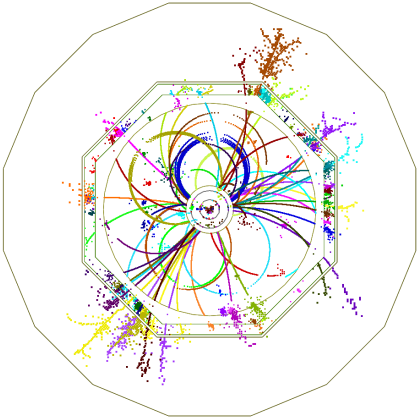
Particle Flow reconstruction at linear colliders

- linear collider detectors are optimized for PFA:
  - high granular calorimeters
  - high hermiticity
  - excellent tracking efficiency

PFA

- reconstruct every single particle
- use tracks for all charged particles
- use Ecal for photons
- use Hcal for neutral hadrons

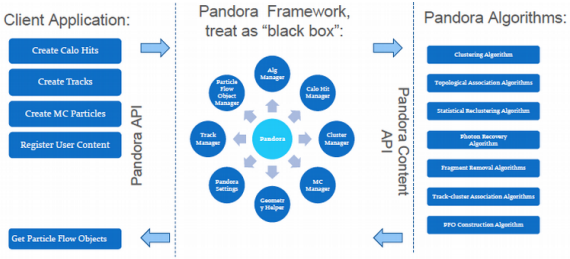
- *trivial* at a B-factory
- not so trivial in multi jet events at a few hundred GeV ...



ttbar @ 500 GeV in ILD

PandoraPFA

- generic framework for pattern recognition in calorimeters
- originally developed for ILC and CLIC (U.Cambridge)
- state of the art particle flow algorithm for highly granular calorimeters
- **AIDA2020** project
  - application to LAr-TPC at neutrino experiments
  - application to HL-LHC
  - implement parallelization



J.Marshall

Possible connection between ILC and Belle-2 software

- common tools used
  - early studies for Belle II done in iLCSoft
  - Belle II has adopted *MarlinKinfit* as *OrcaFit*
  - GBL (Milipede)
  - ...
- in the future:
  - interest in investigating **ACTS** for Belle2
    - to be done in context of AIDA2020
  - LC also looking into this direction
    - => could at least share experience
    - maybe some common developments
  - are there software tools in Belle II that might be of interest to ILC - or HEP in general ?
  - would be nice if Belle II software stack was *open source*

Summary

- **iLCSoft** (`https://github.com/iLCSoft`) is the *software ecosystem* used for linear collider detector studies and beyond
- from the start tried to develop flexible and generic tools that can be applied also to other HEP detector concepts
- recently developed new software tools in AIDA2020:
  - DD4hep, MarlinTrk, PandoraPFA,...
- would like to continue collaboration with other experiments, such as Belle II on common HEP software tools - in the spirit of the HSF

Questions - Comments ?