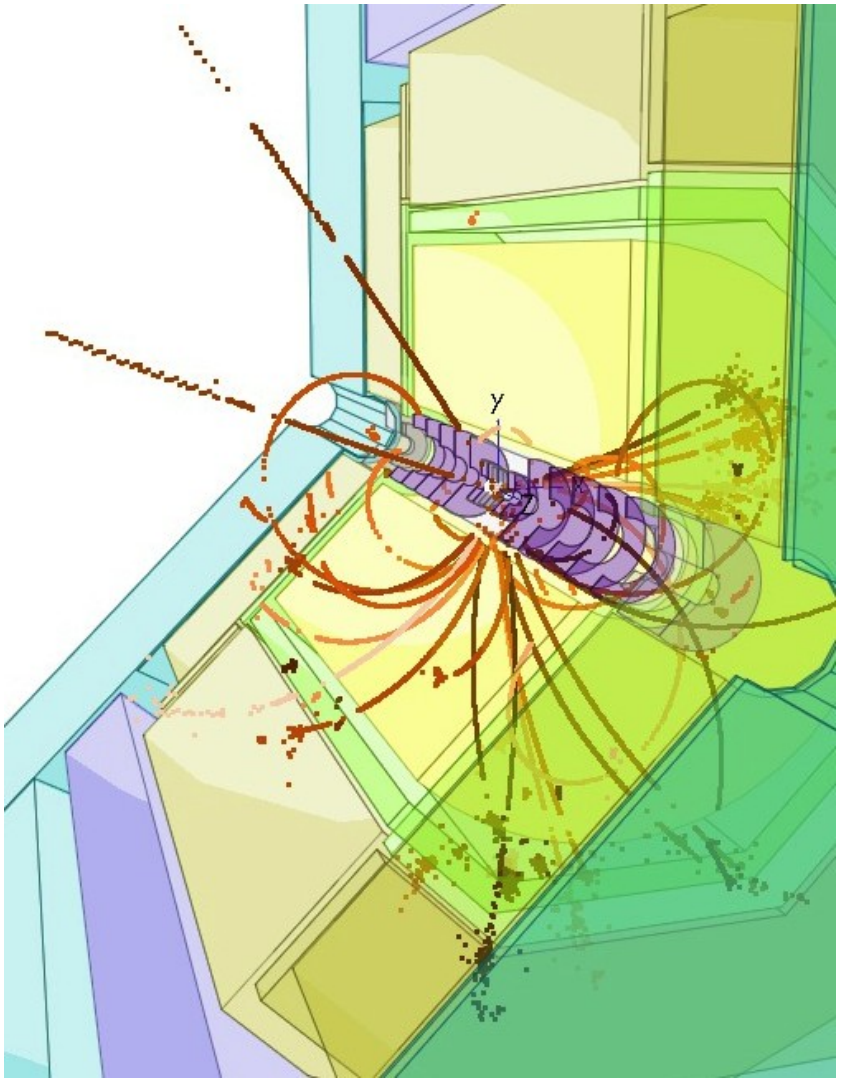
iLCSoft - Status and Future

Frank Gaede, DESY
for the ILD software working group

LCWS 2013
Tokyo, Nov 11-15, 2013

Outline

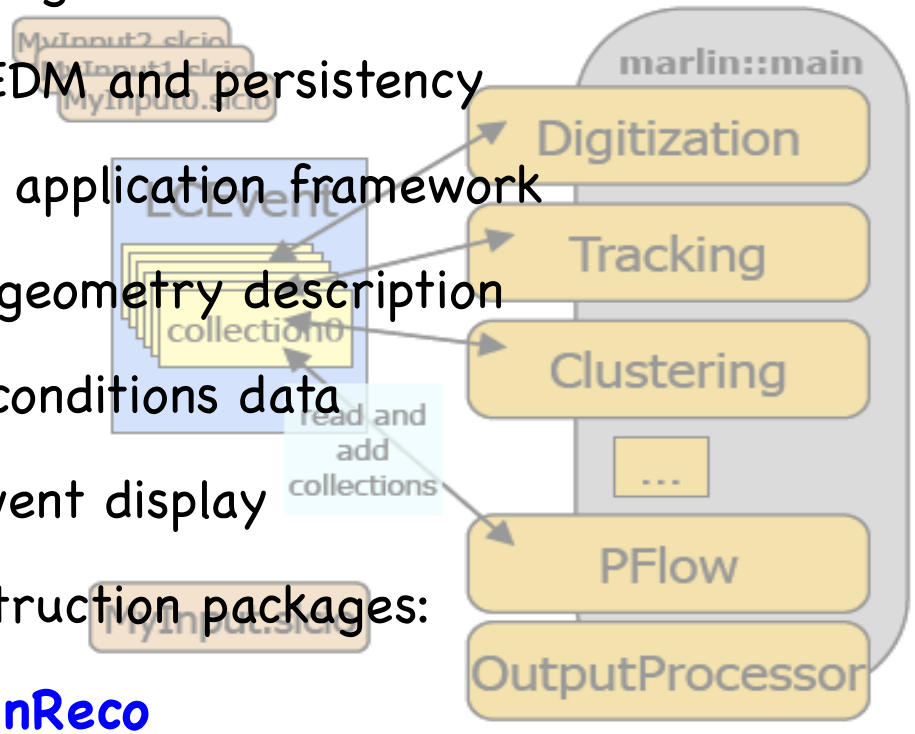
- Introduction to iLCSoft
  - software packages and tools
- iLCSoft Status
- Current activities
  - DD4hep Geometry description
  - Tracking Toolkit
- Summary & Outlook

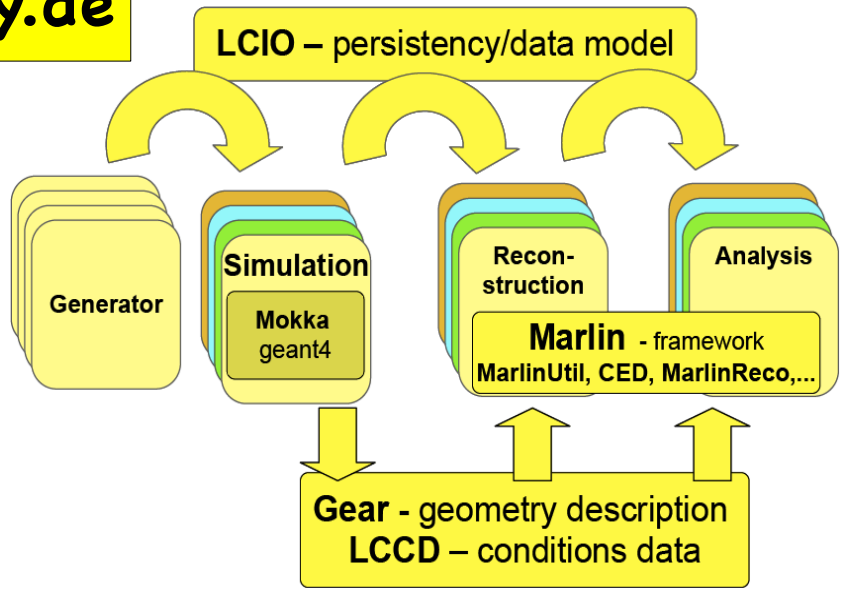


iLCSoft framework - Overview

http://ilcsoft.desy.de

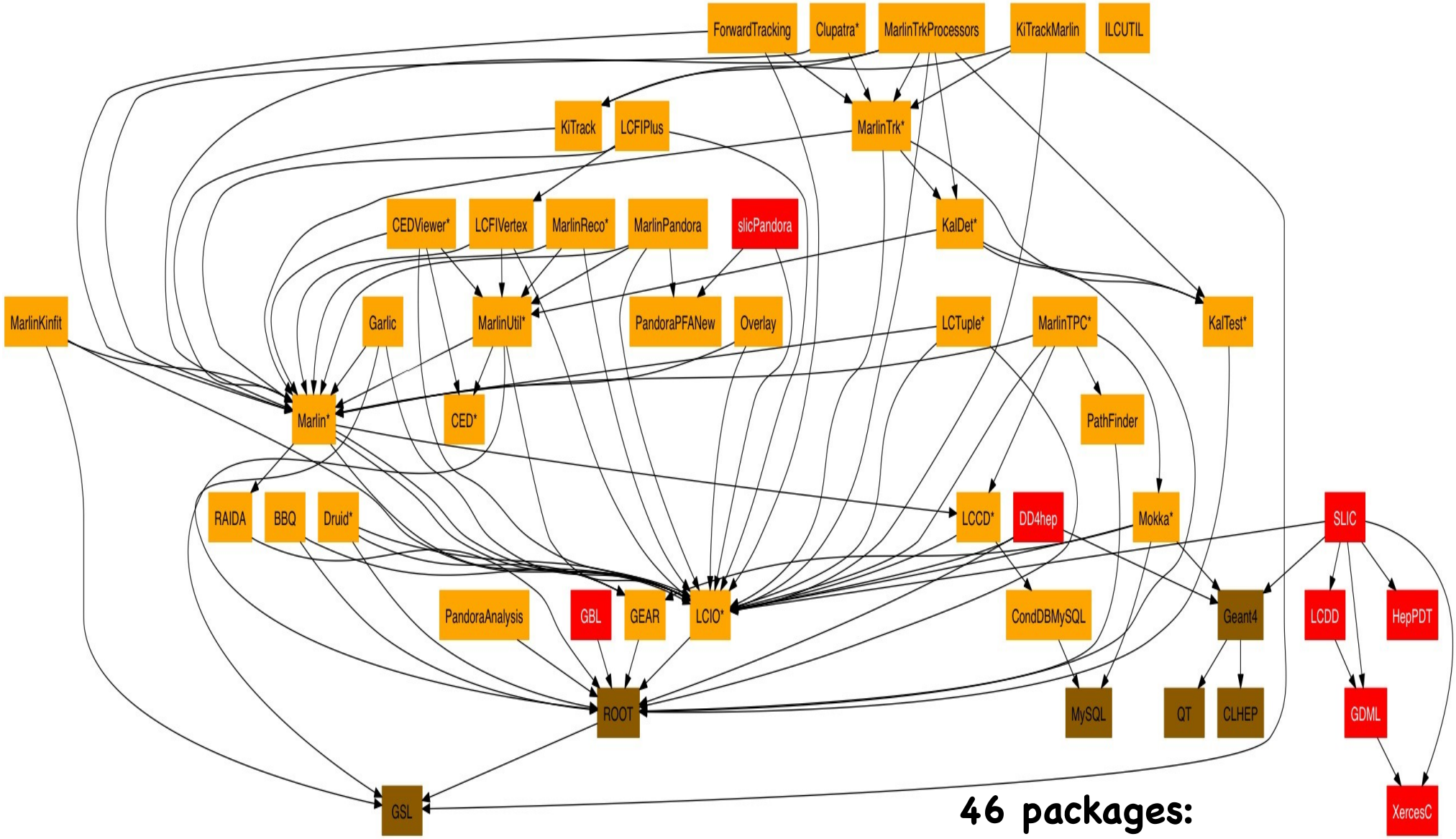
- **Mokka** geant4 simulation - LLR
- **LCIO** EDM and persistency
- **Marlin** application framework
- **GEAR** geometry description
- **LCCD** conditions data
- **CED** event display
- reconstruction packages:
  - **MarlinReco**
  - **MarlinTrk, Clupatra, ForwardTracking,...**
  - **MarlinPandoraPFA**
  - **LCFIVertex, LCFIPlus**
  - **MarlinKinFit**
- many more (see next slide)





- complete sw framework used in Monte Carlo & 'real experiments':
  - **ILD & CLIC** detector concept studies
  - **Calice, LC-TPC, EUTelescope** testbeams
- partly also used by **SiD**

iLCSoft packages (release v01-17-04)



46 packages:

6 external

8 added since LCWS2012

iLcSoft: role-out & installations

- provide reference installations in afs for usage from anywhere on ScientificLinux and compatible platforms:

/afs/desy.de/project/ilcsoft/sw/_OS_/v01-17-04

_OS_: **x86_64_gcc41_sl5** # i686 CPU, 64 bit, gcc4.1, SL5 and compatible
x86_64_gcc44_sl6 # i686 CPU, 64 bit, gcc4.4, SL6 and compatible

- plan to use ***cvmfs*** file system with identical copy of reference installations for Grid production
  - cooperation CERN/DESY

- use ilcinstall tool for your own installation
  - (release tag == ilcsoft version)

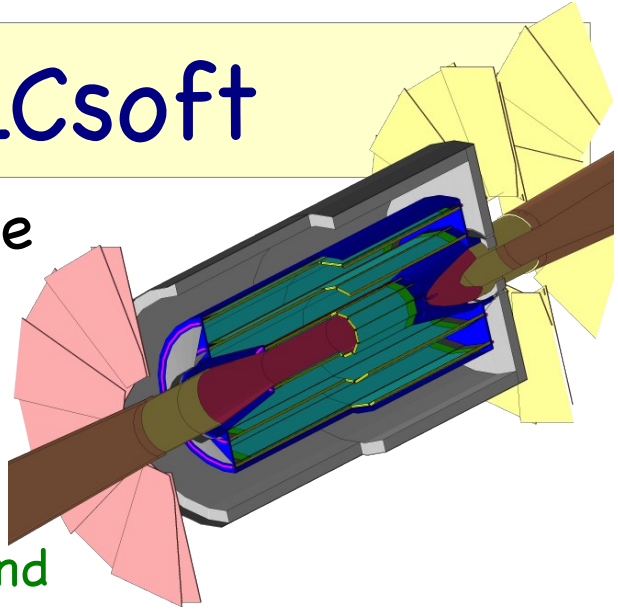
https://svnsrv.desy.de/viewvc/ilctools/ilcinstall/tags/v01-17-04

iLCSoft - software status

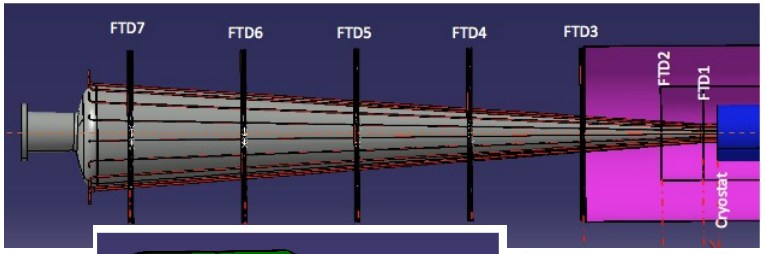
- the timeline for iLCSoft developments in last years mainly driven by the requirements for the (ILD) DBD
  - successfully used to produce O(10^7) fully simulated and reconstructed Monte Carlo Events

- **improved/adapt core tools**
  - LCIOv2, GEAR, CED,...
- **improved realism of the simulation**
  - include gaps, imperfection and services - (ILD_oX_v05 models)
- **complete re-write of tracking code**
- improvements and re-write of reconstruction algorithms **PandoraPFA** and **LCFIVertex/LCFIPlus**
- developed and used **GridProductionSystem**

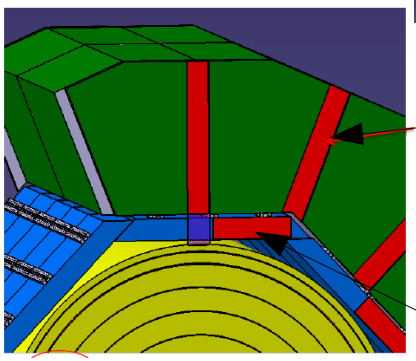
Mokka simulation in iLCsoft



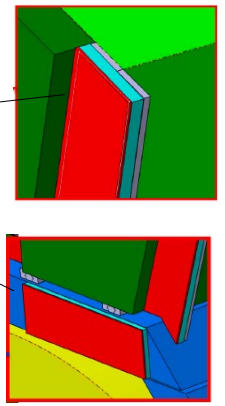
- considerable effort invested by ILD to increase the level of detail and the realism in the ILD simulation models:
  - gaps between sensitive modules
  - added correct dead material for electronics, cables and services (power and cooling)
  - singles wafers and petals for Si-trackers
- simulation models in close agreement with engineering model



ILD_O1_v05 "ILD model using **AHCal**"
ILD_O2_v05 "ILD model using **SDHCal**"
ILD_O3_v05 "ILD model using **SciW Ecal** and **AHCal**"



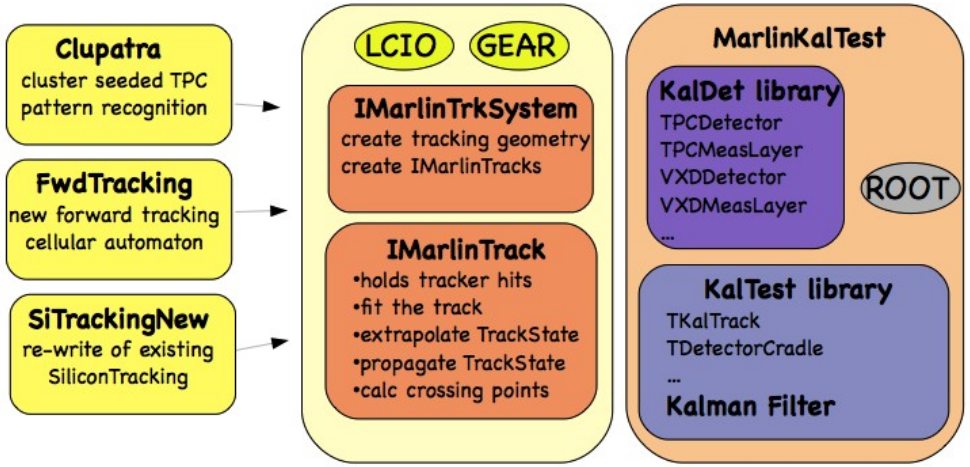
cooling pipes



electronics

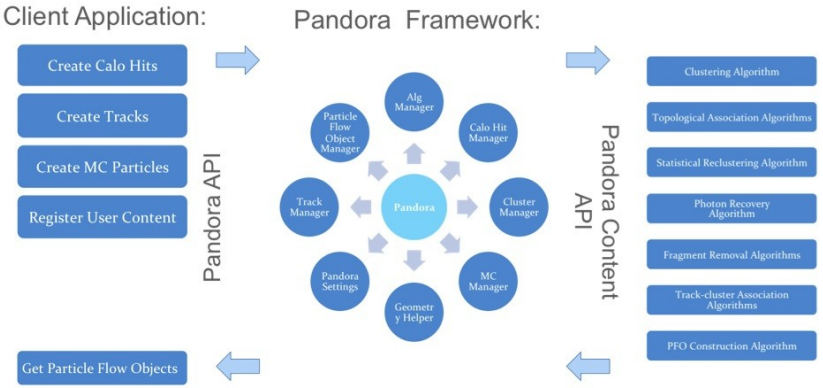
- in the future there will be only limited support for Mokka
- -> need replacement on mid term time scale !

reconstruction tools in iLCSoft

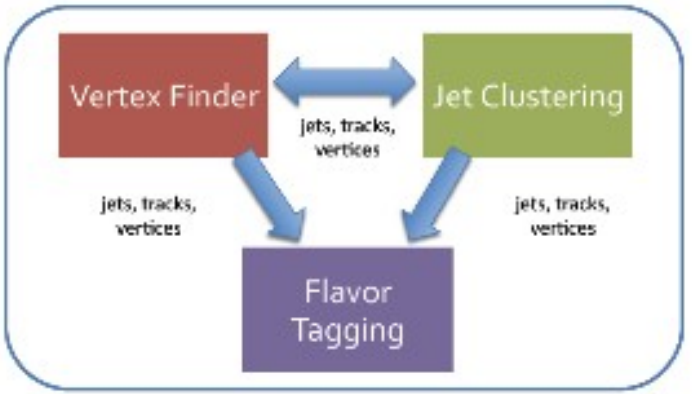


- ILD: new tracking: **MarlinTrk**
  - TPC patrec - Clupatra
  - fwd patrec - ForwardTracking
  - re-write of SiTracking and FullLDC
- replacing old LEP f77 tracking

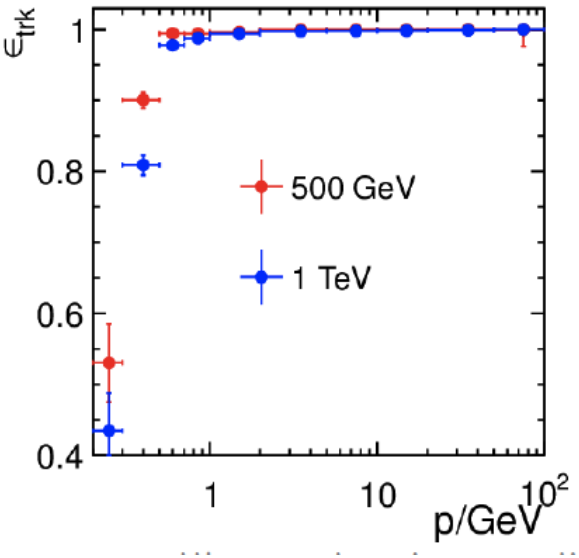
- **PandoraPFA**
  - already re-designed and massively improved for CLIC CDR
  - used by ILD and SID with small adaptations

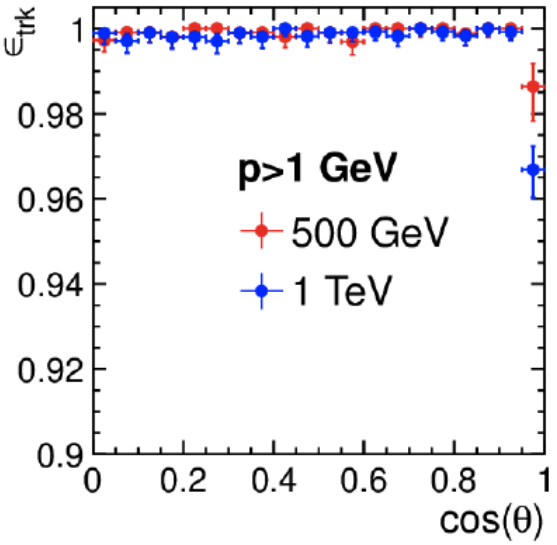


- **LCFIPlus**
  - new improved algorithms for vertexing and flavor tag (based on LCFIVertex)
  - also used for both SID and ILD

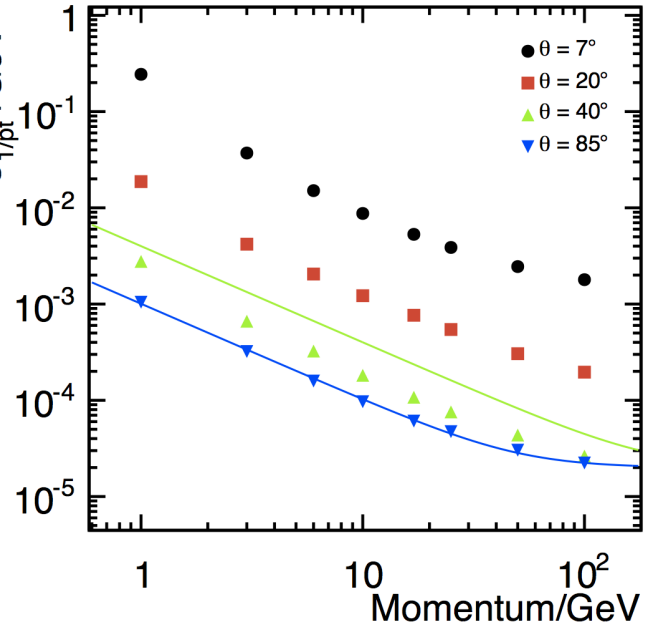


iLCSoft reconstruction performance

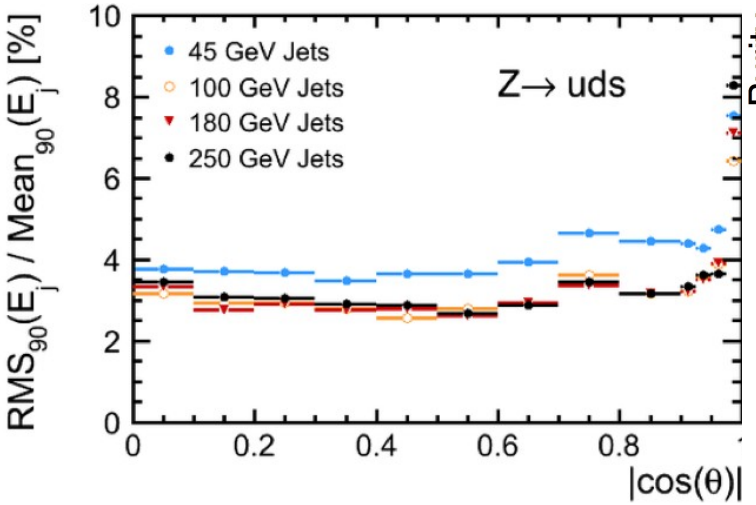


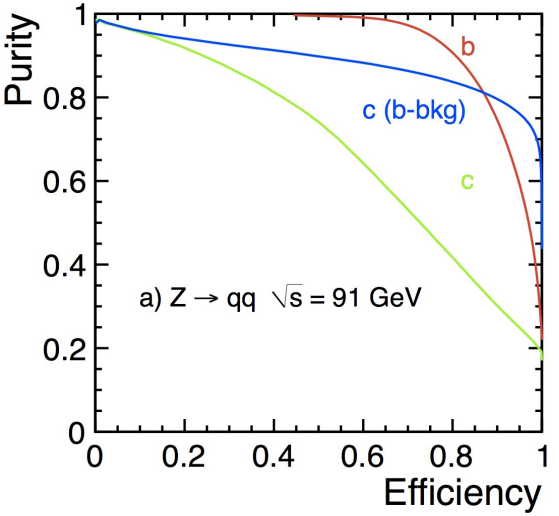


some examples of the software (*and detector*) performance for ILD with current iLCSoft tools



- the simulation and reconstruction tools in iLCSoft have reached a considerable level of maturity
- used to demonstrate that the design goals for ILD detector performance are met





iLCSoft activities after DBD

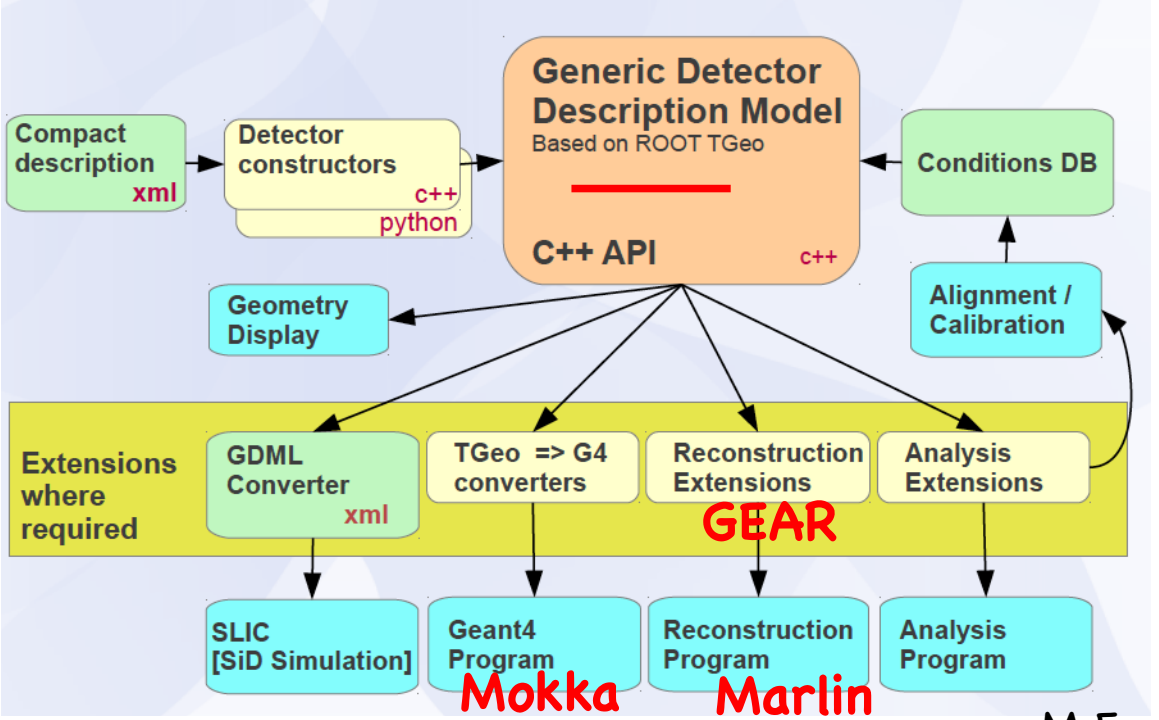
- started series of developers releases v01-17-0x
  - current v01-17-04 - v01-17-05 before end of the year
  -
- new packages added in v01-17-0x:
  - GBL: General Broken Lines (C.Kleinwort, Ch.Rosemann)
  - DD4hep: Geometry description (M.Frank, P.Mato, C.Grefe, FG)
  - SLIC, XercesC, HepPDT, GDML, LCDD, slicPandora (J.McCormick)
- new features added in v01-07-0x series:
  - added **python bindings to LCIO** (C.Grefe)
  - **partial reading of LCIO files** (considerable I/O speed improvement)
  - **pyced**: interactive version of CED event display (t.b.r in v01-17-05)
  - port Clupatra to work w/ real test beam data
  - CED autoshot feature (A.Miyamoto)

towards common LC software

- general agreement that only way forward is to move to common software tools
- goals of closeout of 2012 LC-Software Meeting:
  - a common simulation application based on the geometry description developed in AIDA WP2
  - a common C++ tracking package in the context of AIDA WP2
- at meeting in 01/2013 discussion focused on the details of how these goals can be achieved
  - interface between geometry description and simulation
  - interface to reconstruction (tracking)
  - -> decision to develop prototypes to investigate options
- agreement to use DD4hep as geometry tool

DD4hep - overview

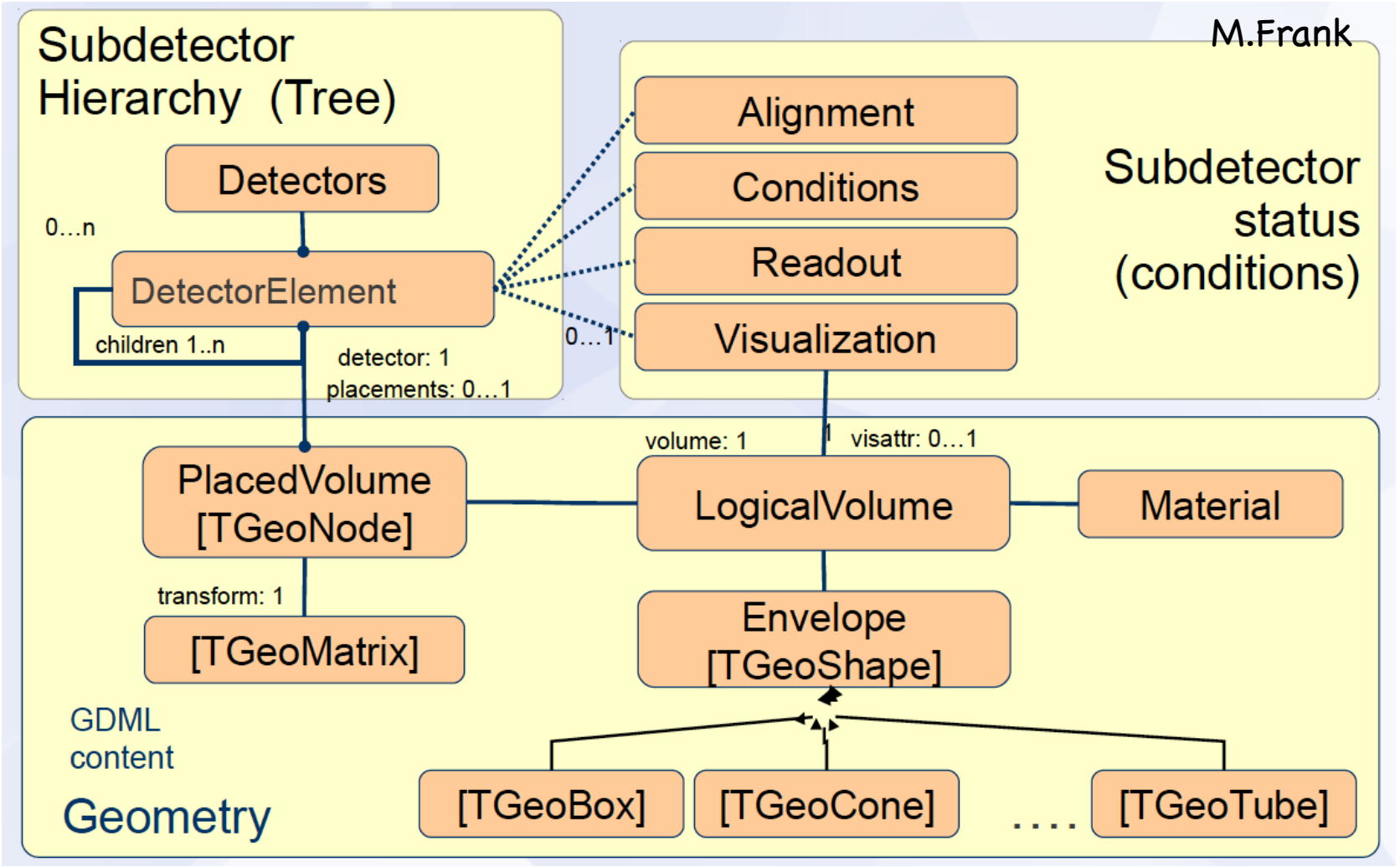
- goals for DD4hep:
  - full detector description
  - full experiment life cycle
  - consistent description
  - ease of use
- DD4hep is based on best concepts from existing geometry tools:
  - xml files with parameters and compact description
  - C++ drivers per sub detector
  - provide several interfaces to simulation, reconstruction and analysis programs



M.Frank

- **DD4hep** common detector geometry description
  - developed by CERN-SFT/LDC (+DESY) in AIDA WP2
- midterm goal: all concept working groups use **SLIC** simulation

DD4hep geometry implementation



- DD4hep uses the TGeo geometry classes to instantiate geometry tree -> can use all TGeo features directly
- additional (user) code added to detector element class

DD4Hep - xml description

- inspired by compact detector description from SiD software
- human readable and compact geometry description in XML format
- used as the main input to the detector description system
- extendable with new generic detector types
- xml parsing mostly done by DD4Hep core - users just specify data structures for DetectorElements

```
- <detector name="VXD" type="ILDExVXD"
  vis="VXDVis" id="1" readout="VXDHits">
  - <layer id="1" vis="VXDLayerVis"
    phi0="-1.570796327e+00">
      <support thickness=".1*mm" material="Carbon"
      vis="VXDSupportVis"/>
      <ladder zhalf="65*mm"
      radius="1.595000000e+01*mm"
      width="1.100000000e+01"
      offset="-1.874869853e+00*mm"
      thickness="0.05*mm" material="Silicon"
      number="10"/>
    </layer>
  - <layer id="2" vis="VXDLayerVis"
    phi0="-1.570796327e+00">
      <support thickness=".1*mm" material="Carbon"
      vis="VXDSupportVis"/>
      <ladder zhalf="65*mm" radius="18*mm"
      width="1.100000000e+01"
      offset="-1.874869853e+00*mm"
      thickness="0.05*mm" material="Silicon"
      number="10"/>
    </layer>
    <!-- ... -->
  </detector>
```

Extension of the geometry system

- the DetectorElement can be extended with user code
- in particular useful for reconstruction ("GEAR++")
- any high level view on the detailed geometry can be coded
- user program does not need to know the exact details of the volumes
- the details of the detector geometry can change without the need for changing the reconstruction or analysis code

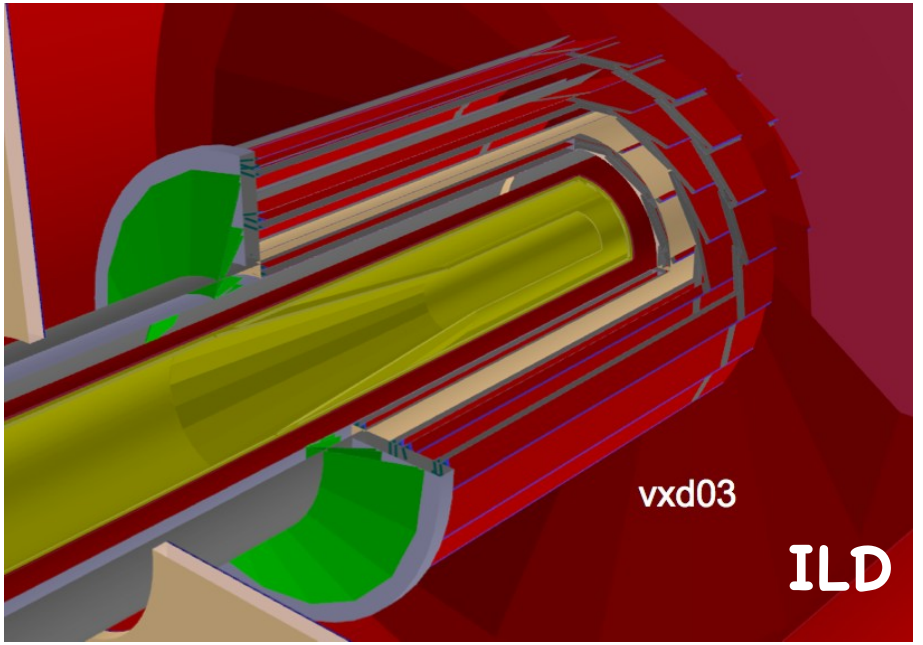
```
struct GearTPC : public Geometry::Subdetector {
  typedef TPCData Object;
  GearTPC(const Geometry::RefHandle<TNamed>& e);

  GlobalPadIndex  getNearestPad (double c0, double c1) const;
  double getDriftVelocity () const;
  double getReadoutFrequency () const;
  double getInnerRadius() const;
  double getOuterRadius() const;
};
```

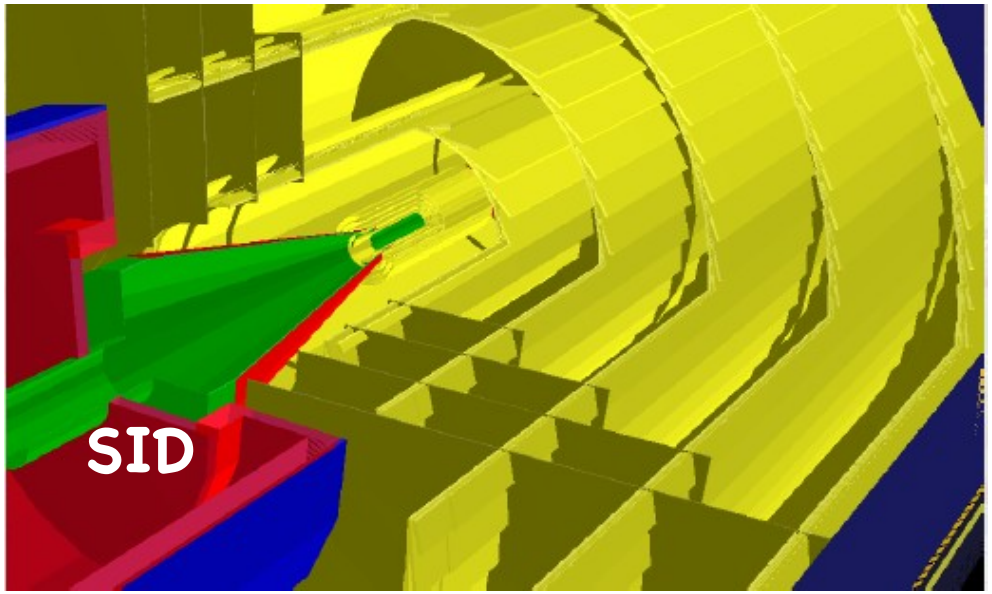
```
double GearTPC::innerRadius() const {
  Subdetector gas   = data<Object>()->gas;
  Tube        tube  = gas.volume().solid();
  return tube->GetRmin();
}
double GearTPC::outerRadius() const {
  Subdetector gas   = data<Object>()->gas;
  Tube        tube  = gas.volume().solid();
  return tube->GetRmax();
}
```

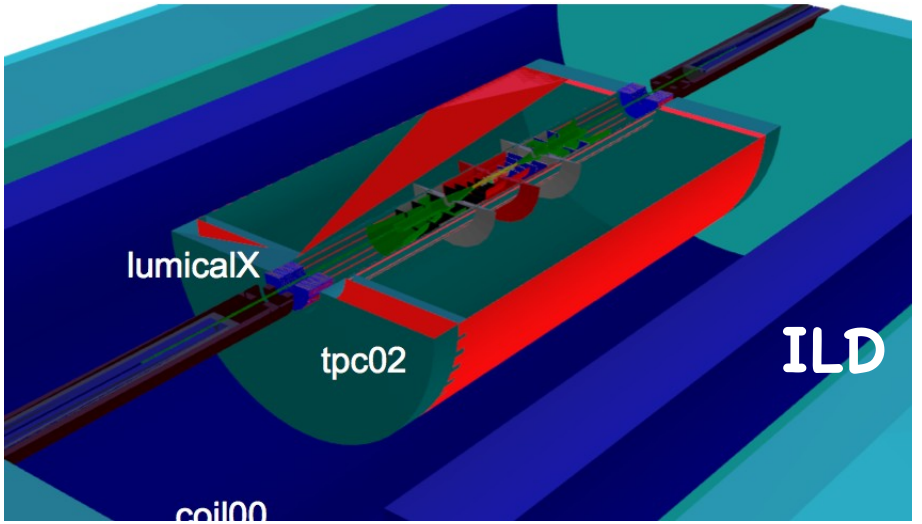
*DetectorElement previously called SubDetector

DD4hep detector visualization

- displaying the detector geometry works via TGeo or gdml:
- ROOT OGL viewer, TEve, ...

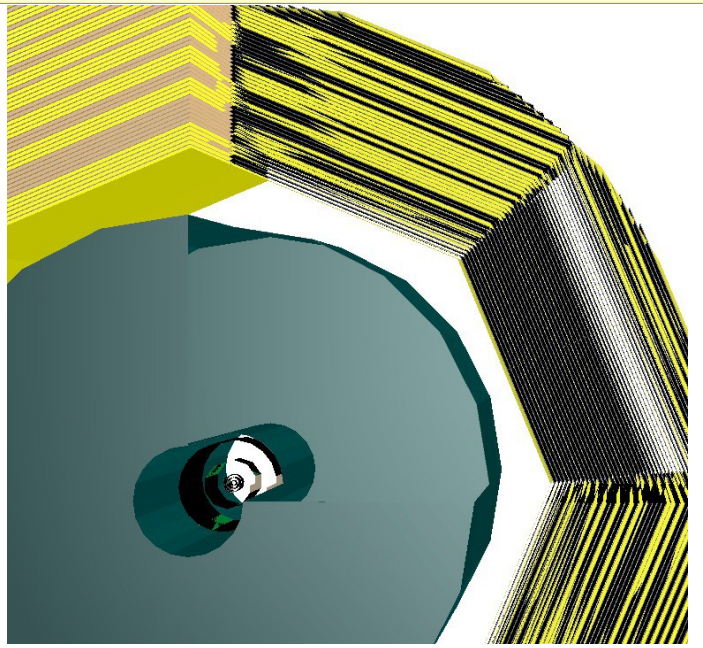


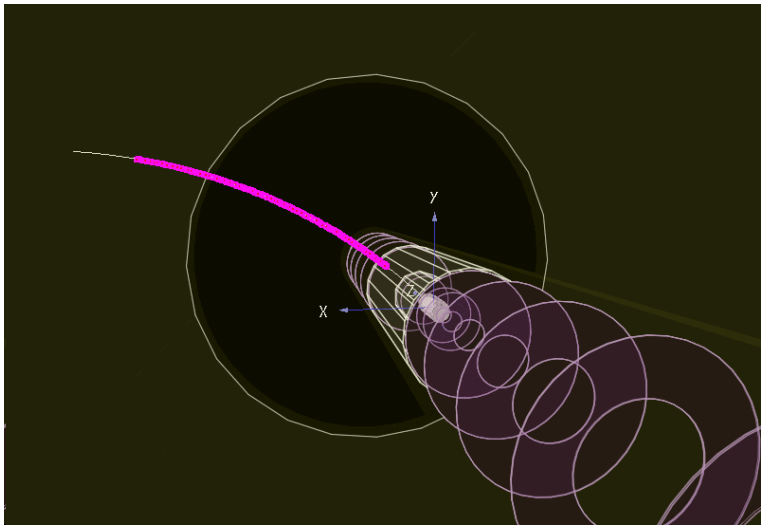




DD4hep status

- developed simple prototypes:
  - ILDExDet: VXD, SIT, TPC, AHcal
  - CLICSiD
  - Calice test beam
- use to study technical issues:
  - cellIDs, detector segmentations, sensitive detectors
  - interface to reconstruction:
    - use GEAR classes
    - develop new reco-API
- latest release:
  - v00-05 in ilcsoft v01-17-05
  - restructured examples and build system
  - added geant4-example: noviceN04

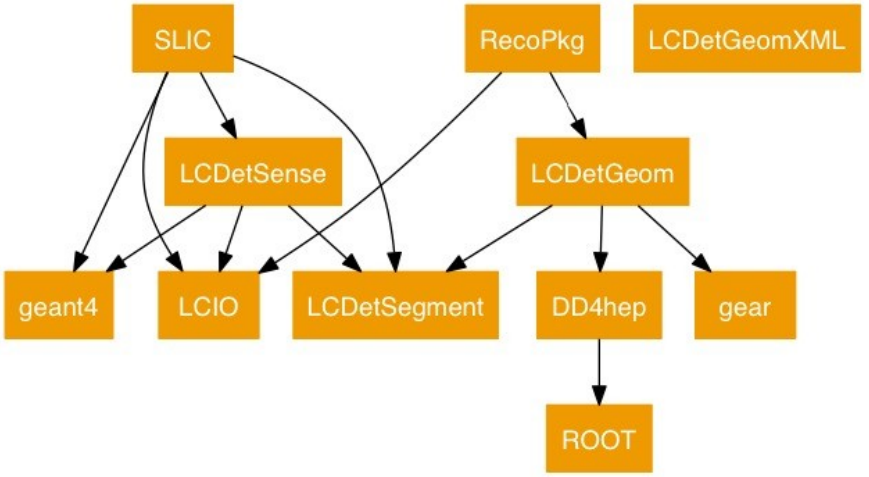


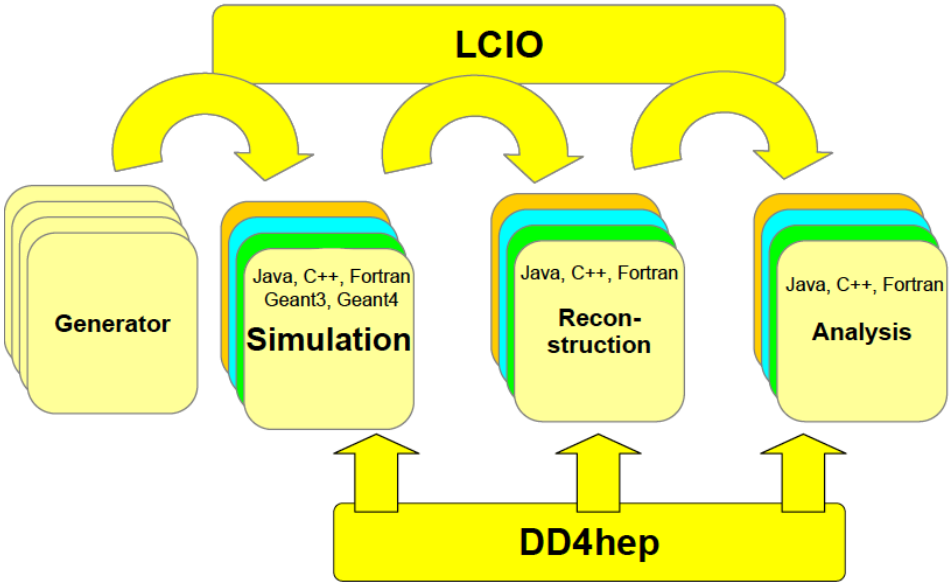


single muon simulated w/ ILDExDet
reconstructed w/ MarlinTrk
*shown in ILD_o1_v05 display

towards using DD4hep

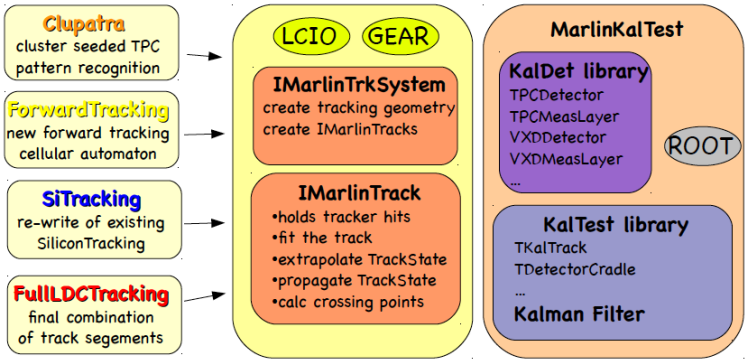
- the goal is to move towards DD4hep as only source of geometry description
- and eventually use **slic** as a common simulation program
- considerable work has gone into the geometry description (SubDetectorDriver) of the current ILD (and Calice) models in Mokka
- need to preserve this work
  - possibly by finding intermediate solution

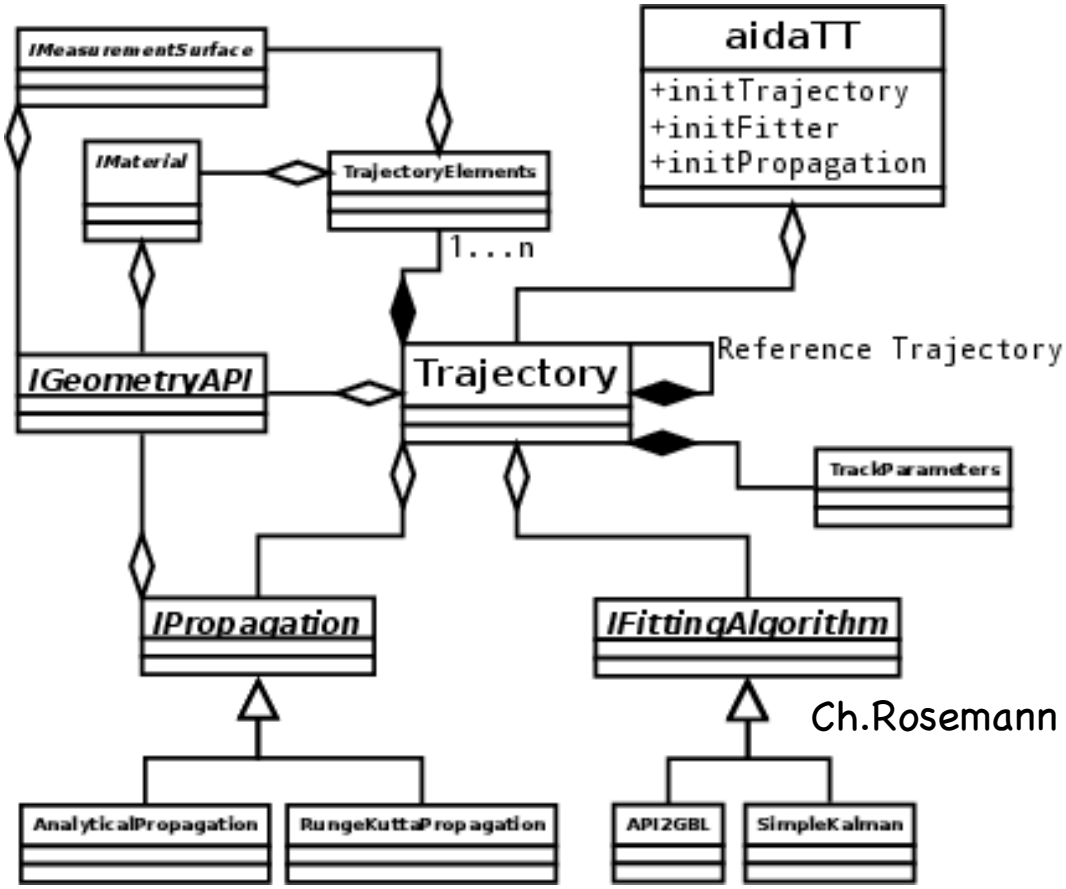




final goal: more flexible and consistent software chain

aidaTT - tracking toolkit





- develop a generic track fitting toolkit in context of AIDA WP2
- based on ideas developed in MarlinTrk and GenFit
- allow to transparently use Kalman-Filter or GeneralBrokenLines
  - needed e.g. for Millipede alignment tool

- next step: define reco interface to DD4hep:
  - measurement surfaces
  - navigation (active and dead material)

Summary & Outlook

- development activities in iLCSoft framework in the last 2-3 years where driven by preparation for the ILD DBD
  - improved realism in simulation and performance of reconstruction tool
- first developers release(s) since then provide new features in core tools and some new packages
- work towards a common simulation and reconstruction framework for Linear Collider studies

- main future activities in iLCSoft framework:
- move to DD4hep and port existing simulation models
- develop more flexible, generic tracking software