Tracking Tools - aidaTT

Frank Gaede, CERN/DESY

AIDA Final Meeting

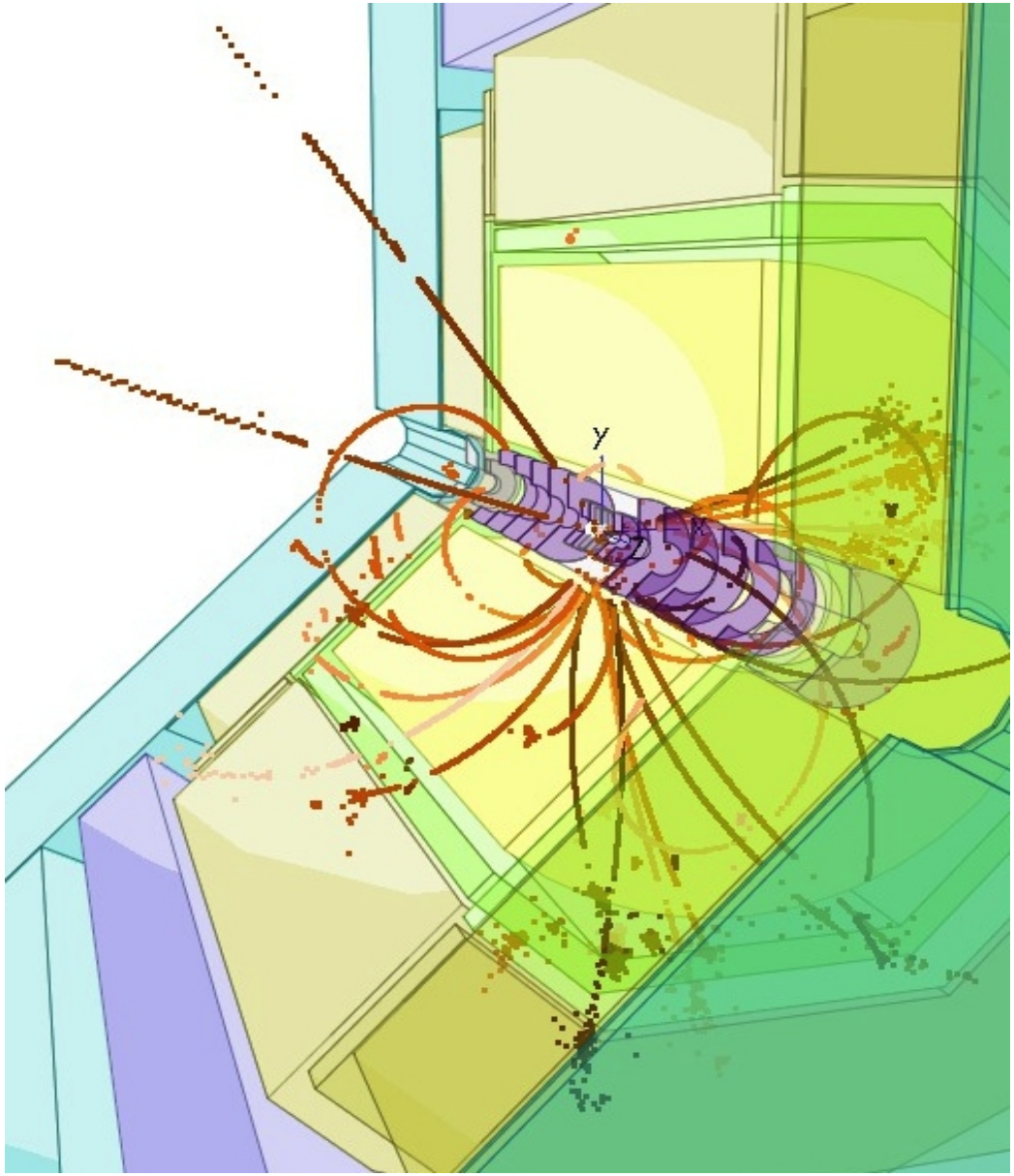
CERN, Dec 9-11, 2014

AIDA is co-funded by the European Commission within the Framework Programme 7 Capacities Specific Programme, Grant Agreement 262025

Outline

- Introduction
- ILD Tracking code
  - Overview
  - Pattern recognition
  - Performance
- aidaTT
  - design
  - status
- Summary & Outlook

people involved
- Steven Aplin, DESY
- Christoph Rosemann, DESY
- Robin Glattauer, OeAW
- FG



Introduction

- Development of a Tracking Toolkit part of sub-task 2.3 of WP2: “Reconstruction Toolkit for HEP” - goals:
- state-of the art track fitting and pattern recognition tools
  - Kalman Filters, General Broken Lines Fits, CA, …
- as much as possible framework and experiment independent code for general use in HEP
- interface to geometry tools developed in task 2.2
- procedure:
- development of the fitting and pattern recognition in the context of the Linear Collider Software framework (iLCSoft)
- immediate application to ILD physics studies and detector R&D
- very loose coupling of tools to the framework so that they can be re-used

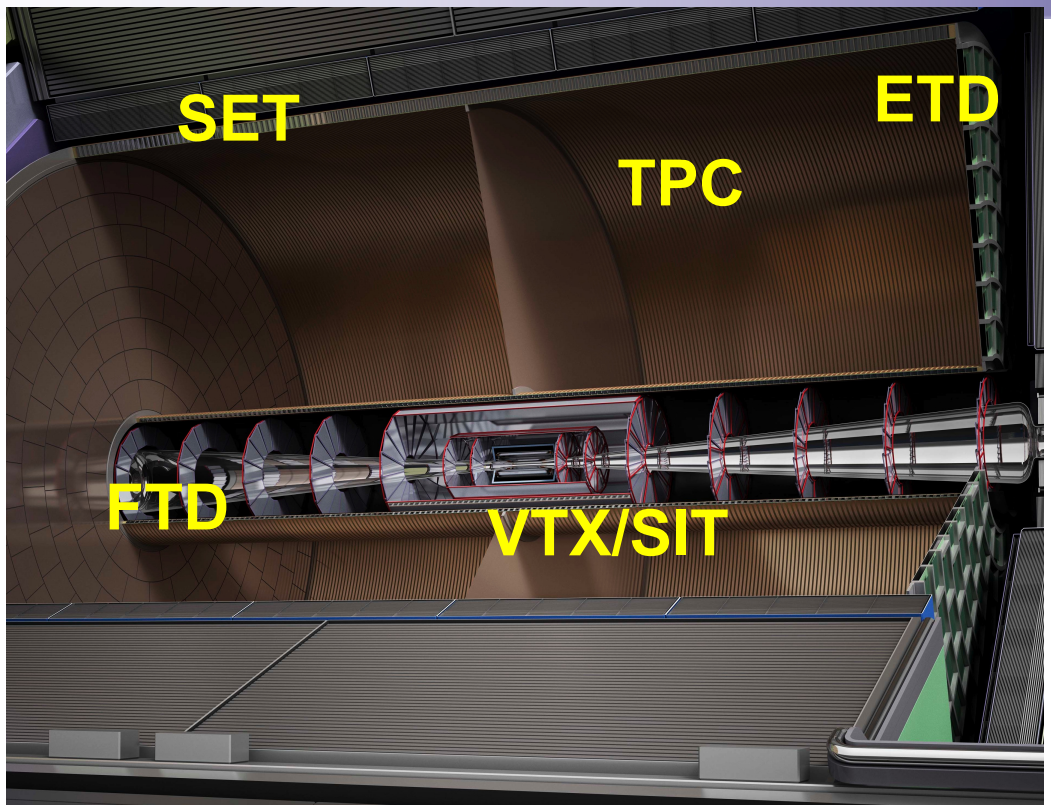
Milestones and Deliverables

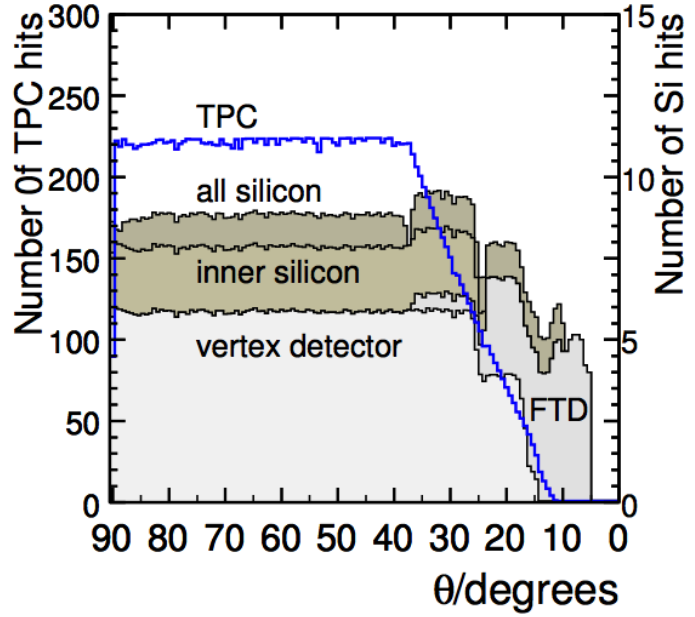
- Deliverables
  - D2.4, M12:Software design for tracking toolkit.
  - D2.8, M44:Software toolkit with tracking algorithms.

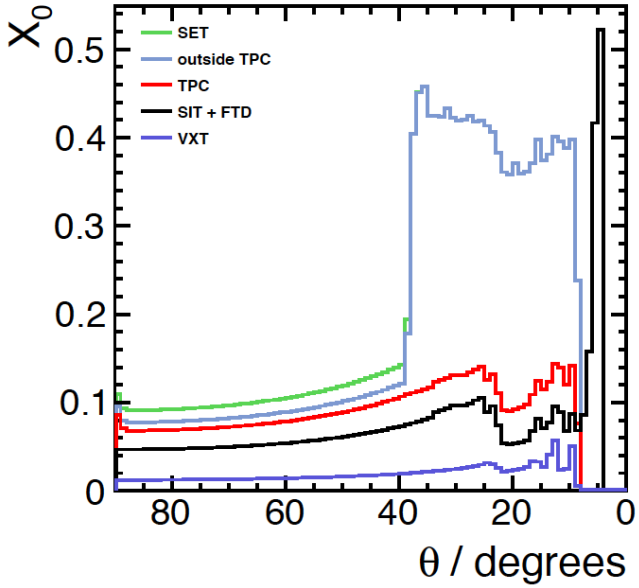
- Milestones
  - MS11, M18:Running prototype of tracking toolkit including some algorithms. Application to ILD-TPC simulation.
  - MS14, M44:Integration of tracking toolkit into LC-software framework. Validation of physics performance.

- Partners:
  - **DESY**
  - **OeAW - Vienna**

The ILD tracking system

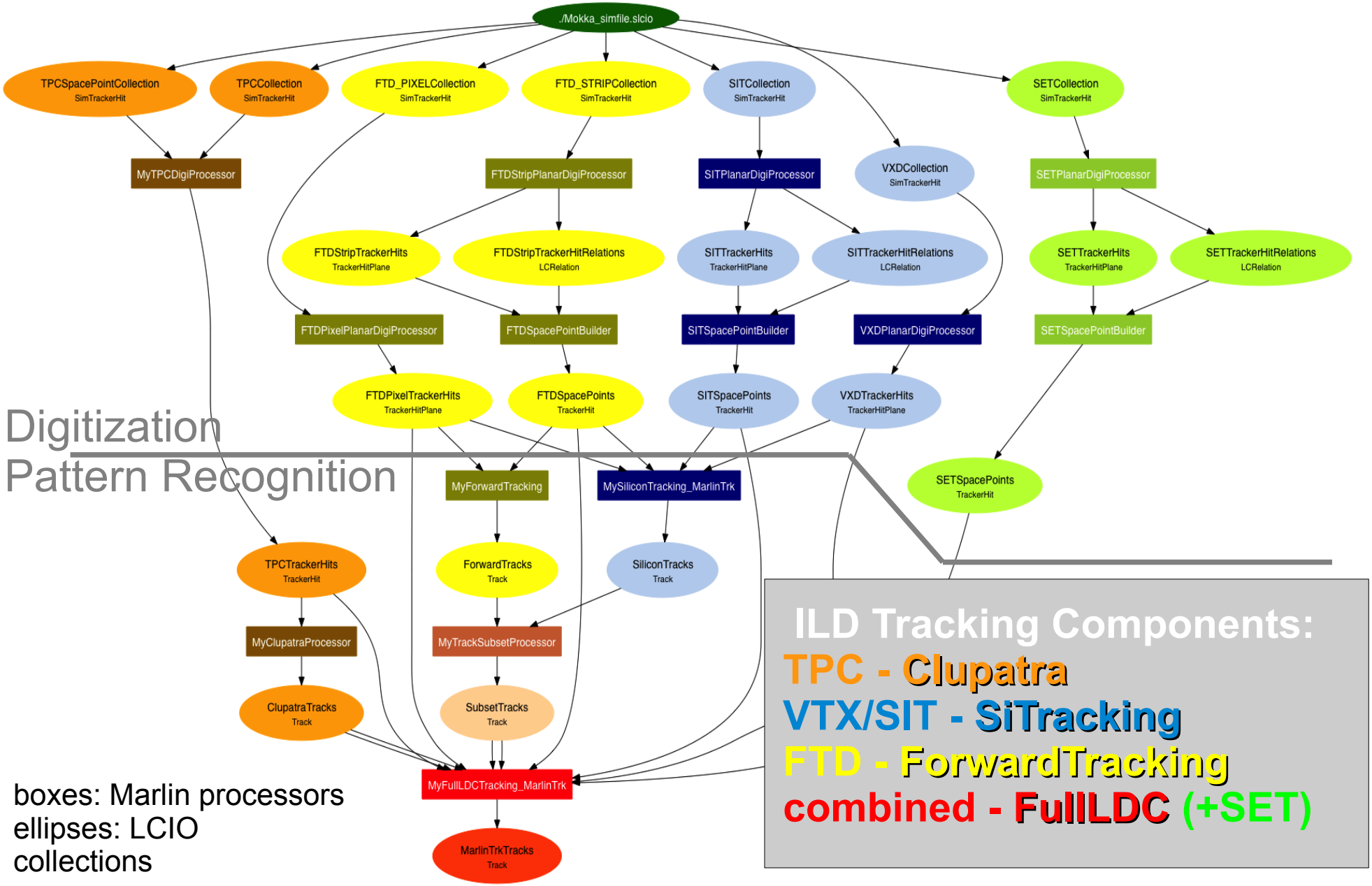






Detector	Point Resolution		
VTX	$\sigma_{r\phi,z}$	=	2.8μm (layer 1)
	$\sigma_{r\phi,z}$	=	6.0μm (layer 2)
	$\sigma_{r\phi,z}$	=	4.0μm (layers 3-6)
SIT	σ_{α_z}	=	7.0μm
	α_z	=	$\pm$7.0° (angle with z-axis)
SET	σ_{α_z}	=	7.0μm
	α_z	=	$\pm$7.0° (angle with z-axis)
FTD *Pixel*	σ_r	=	3.0μm first two discs
	$\sigma_{r_\perp}$	=	3.0μm
FTD *Strip*	σ_{α_r}	=	7.0μm
	α_r	=	$\pm$5.0° (angle with radial direction)
TPC	$\sigma^2_{r\phi}$	=	$(50^2 + 900^2 \sin^2\phi + ((25^2/22) \times (4T/B)^2 \sin\theta)\ (z/\mathrm{cm}))\ \mu\mathrm{m}^2$
	σ^2_z	=	$(400^2 + 80^2 \times (z/\mathrm{cm}))\ \mu\mathrm{m}^2$
	where ϕ and θ are the azimuthal and polar angle of the track direction		

Modules for ILD Tracking



boxes: Marlin processors
ellipses: LCIO collections

ILD Tracking Components:

TPC - Clupatra

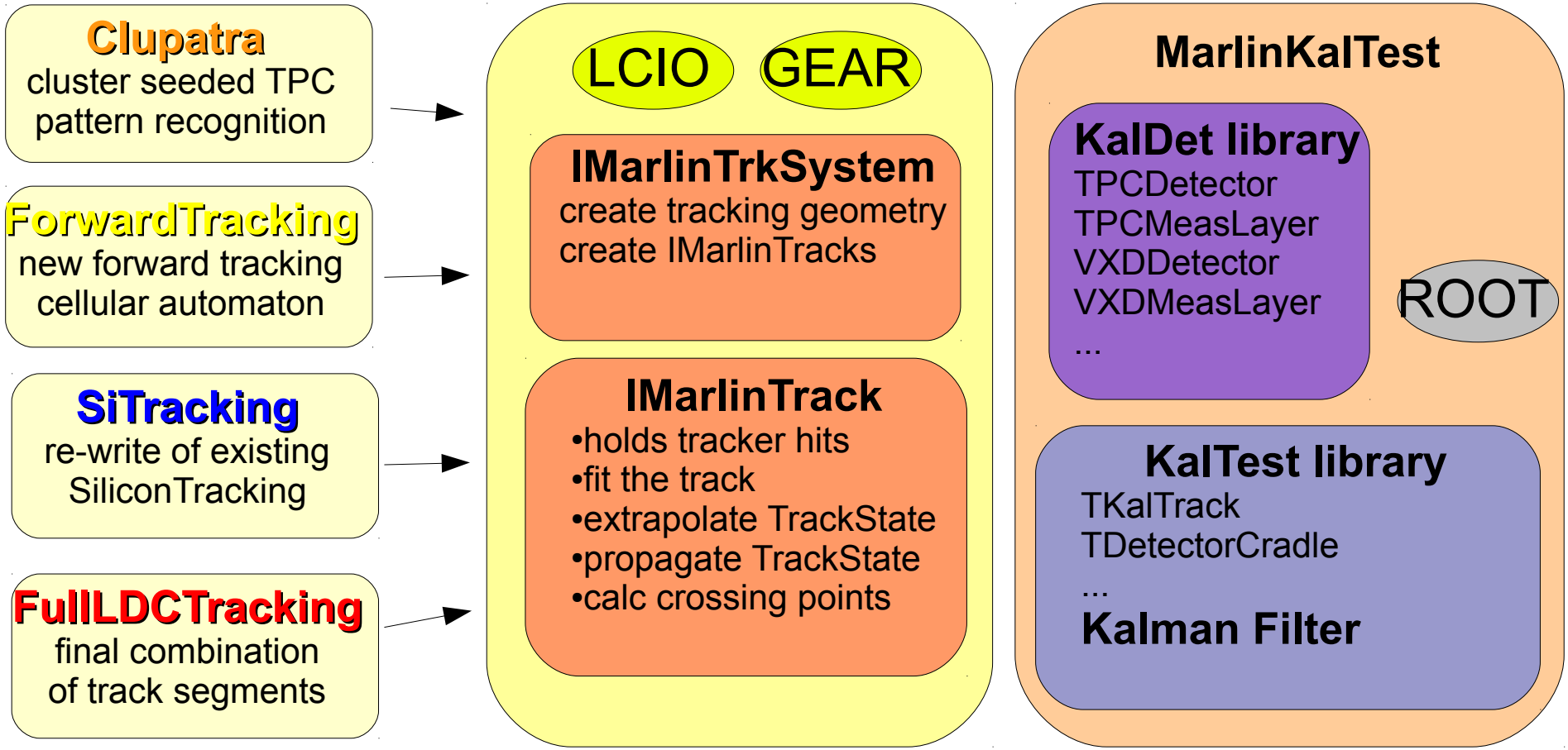
VTX/SIT - SiTracking

FTD - ForwardTracking

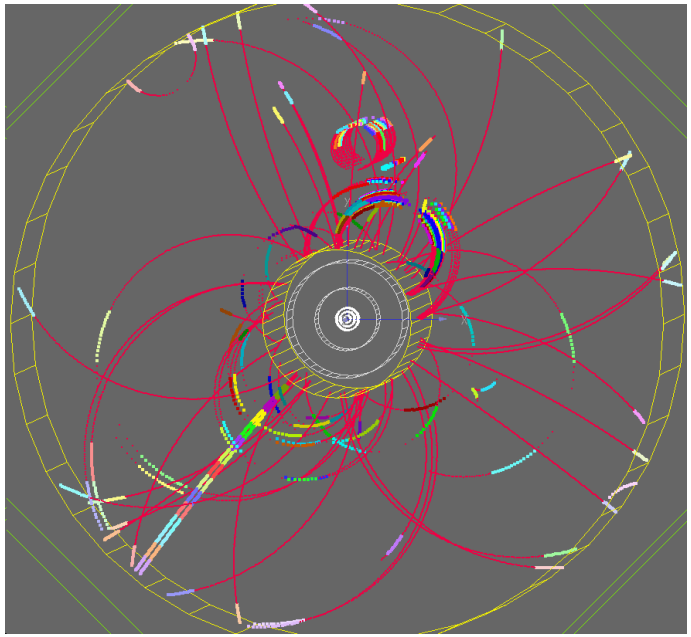
combined - FullLDC (+SET)

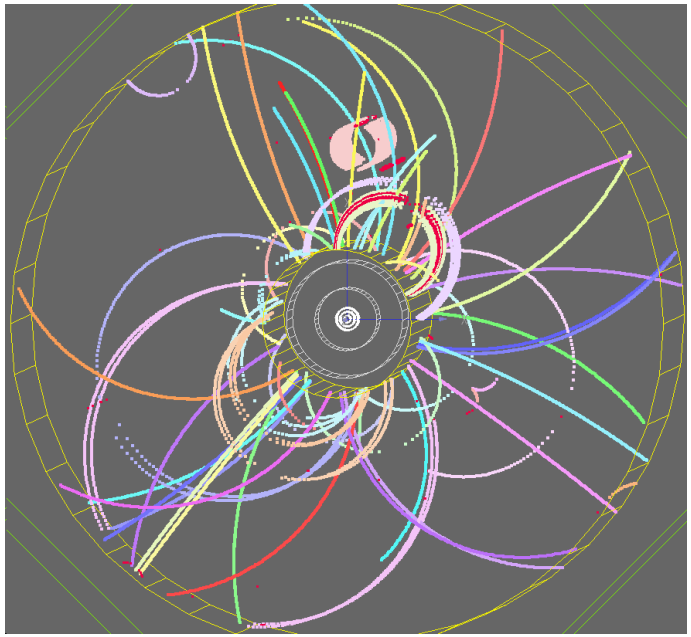
IMarlinTrk interface for tracking

- common API for developing tracking code (TPC, Silicon, Fwd)
- provides loose coupling between pattrec and fitting
- defined abstract interface IMarlinTrk and implement using KalTest/KalDet
- serves as **prototype** for *generic tracking package* in AIDA WP2

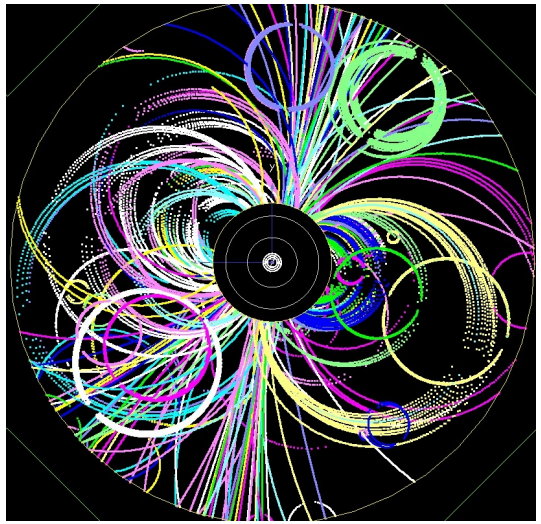


Clupatra: TPC pattern recognition





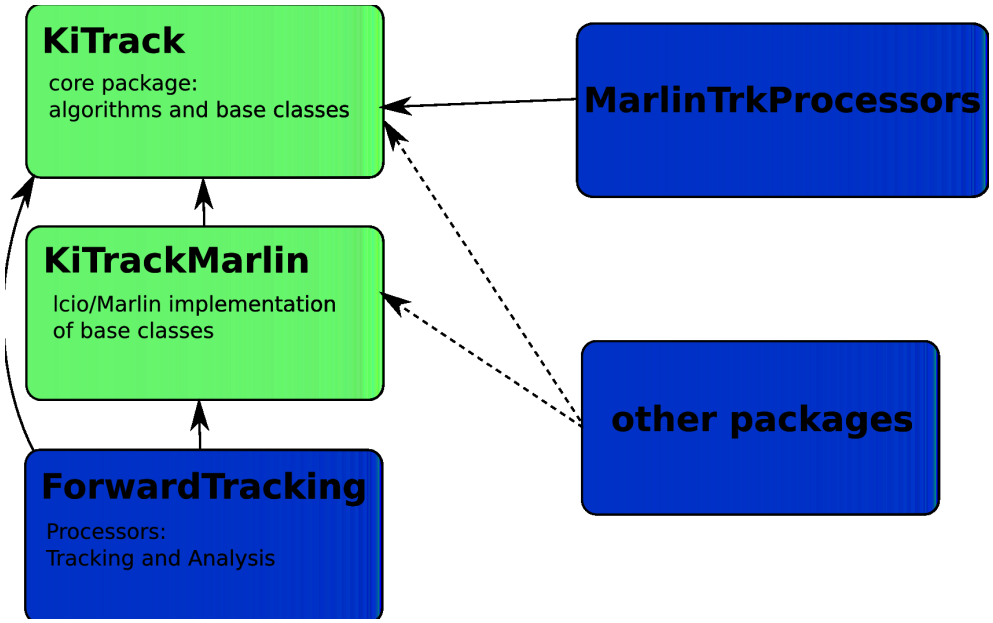
ttbar event @ 500 GeV - ILD

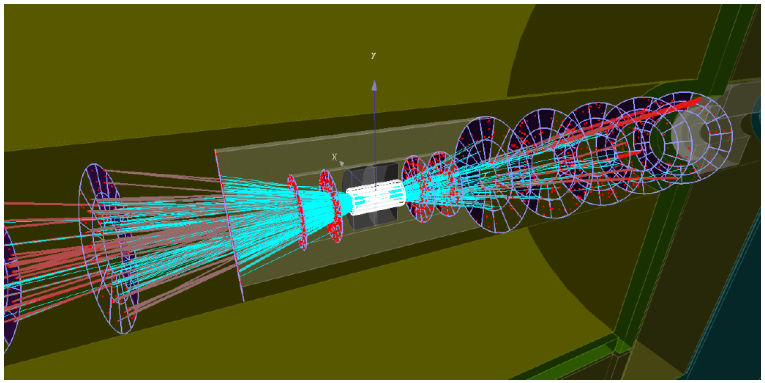


ttbar event @ 3 TeV - CLIC

- track seeding is based on topological nearest neighbor clustering
  - generic template library using space points and distance cuts
- clean track seeds are propagated inwards using the IMarlinTrk interface
- merged seed clusters are split based in hit multiplicity
- track segments from curlers are merged
  - based on rough (O(10%)) criterion for R, delta(xc,yc), tan(lambda)

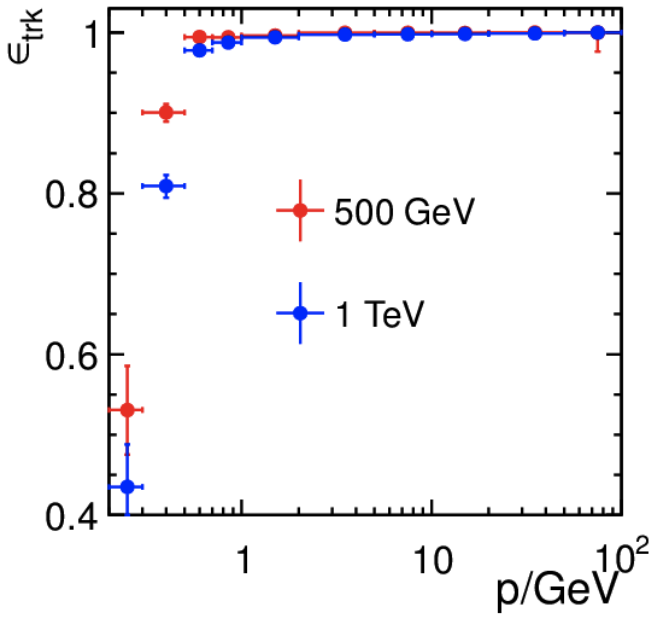
ILD forward tracking

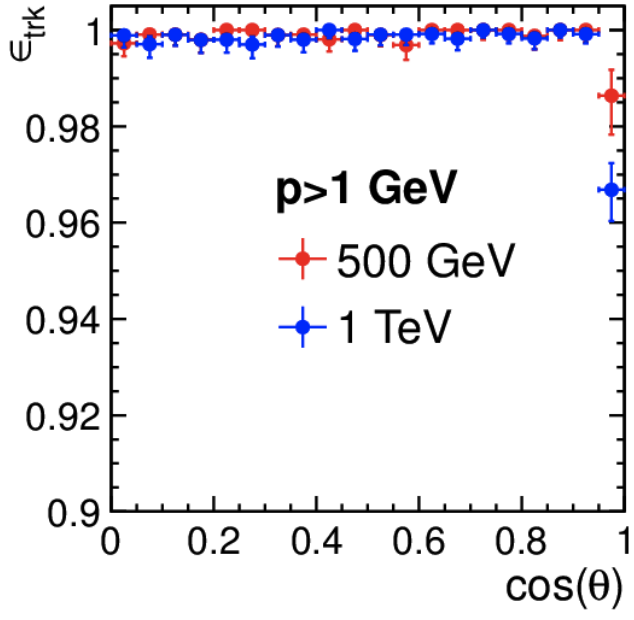


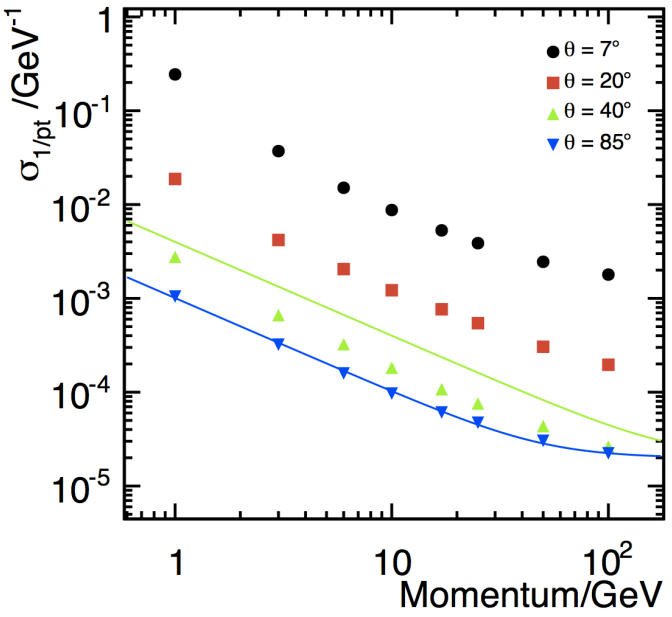


- standalone tracking for forward tracking discs in ILD based on:
- Cellular Automatons for track finding
- Hopfield networks to arbitrate between candidates w/ mutual hits)
- core functionality for CA written in framework independent package KiTrack
- package with base classes for iLCSoft: KiTrackMarlin
  - → allows application to central Si-Tracking
    - currently ongoing for ILD and planned for CLICdp

ILD Tracking Performance







eff_trk > 99.8%, p>1GeV, cos(th)<0.95

$$\sigma_{1/p_T} = \frac{2 \times 10^{-5}}{GeV} \oplus \frac{1 \times 10^{-3}}{p_T \sin\theta}$$

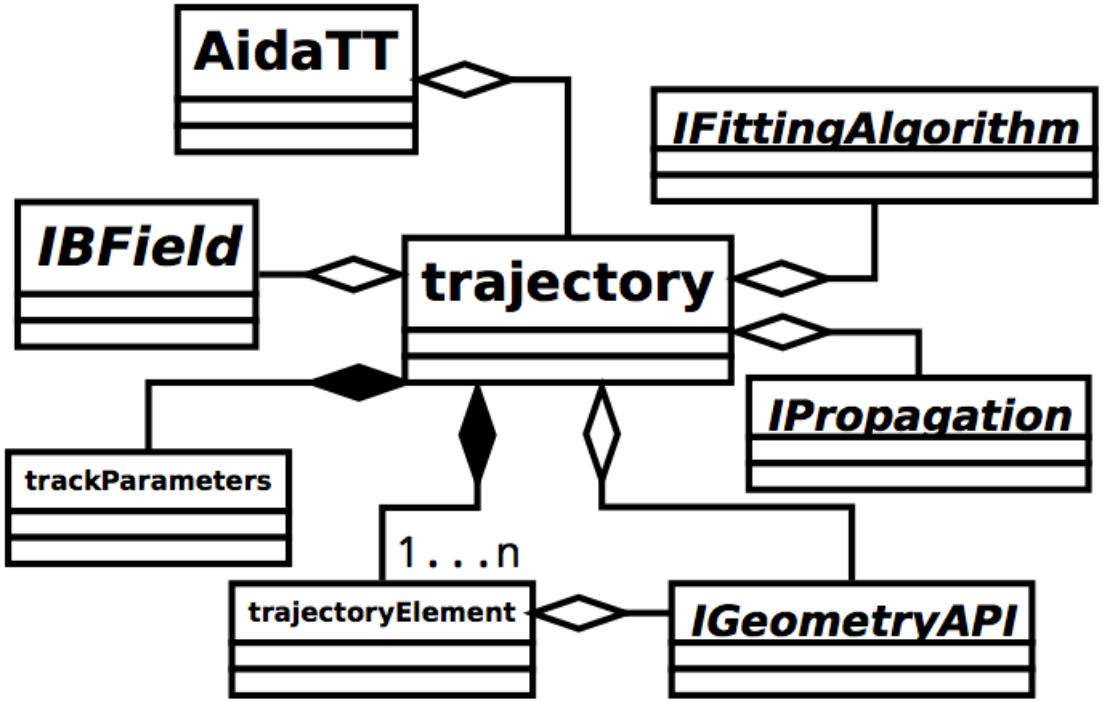
- ILD tracking based on the tools developed in AIDA WP2*
  - Clupatra, FwdTracking, IMarlinTrk

  shows excellent tracking efficiency and transverse momentum resolution
- used for large scale MC production for TDR

*combined w/ pre-existing tools for SiTracking and final track merging

aidaTT - overview

- Tracking Toolkit
  - track fitting (and finding)
    - GBL, Kalman, ...
  - track propagation, extrapolation
  - intersection calculation



- Design
  - completely modular
  - well defined API to reco frameworks
  - separation of data, algorithm and functionality
  - parallelization on single track level possible

central API and implementation

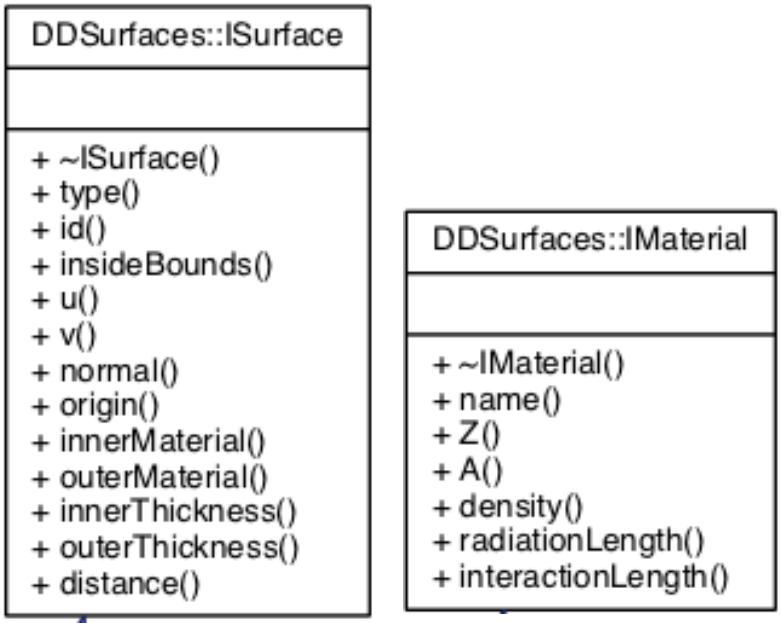
- aidaTT::AidaTT
  - master interface to create aidaTT::trajectory objects instantiates the specific objects: geometry, propagation, fields
- trajectory
  - created/configured with aidaTT::AidaTT
  - holds a set of track states
  - 23 parameters: 5 + 15 + 3
  - allows adding/removing points/elements from the trajectory provides methods for:
    - extrapolation (no material effects)
    - propagation (including material effects)
    - intersecting with basic surfaces

central API and implementation

- trackParameters
  - data class to store the 23 parameters
    - used in external and internal interface
  - additional helper class allows usage of different parameterizations
  - currently implemented L3 (LCIO) perigee parameters:
    - omega, D0, Phi0, Z0, tanL
- trajectoryElement
  - objects assigned to trajectory:
  - hits and intersections with surfaces
    - identified by arc length, wrt reference point of trajectory
    - holds Jacobian to next element
    - measurement information: hit coordinates and measurement surface
    - material information

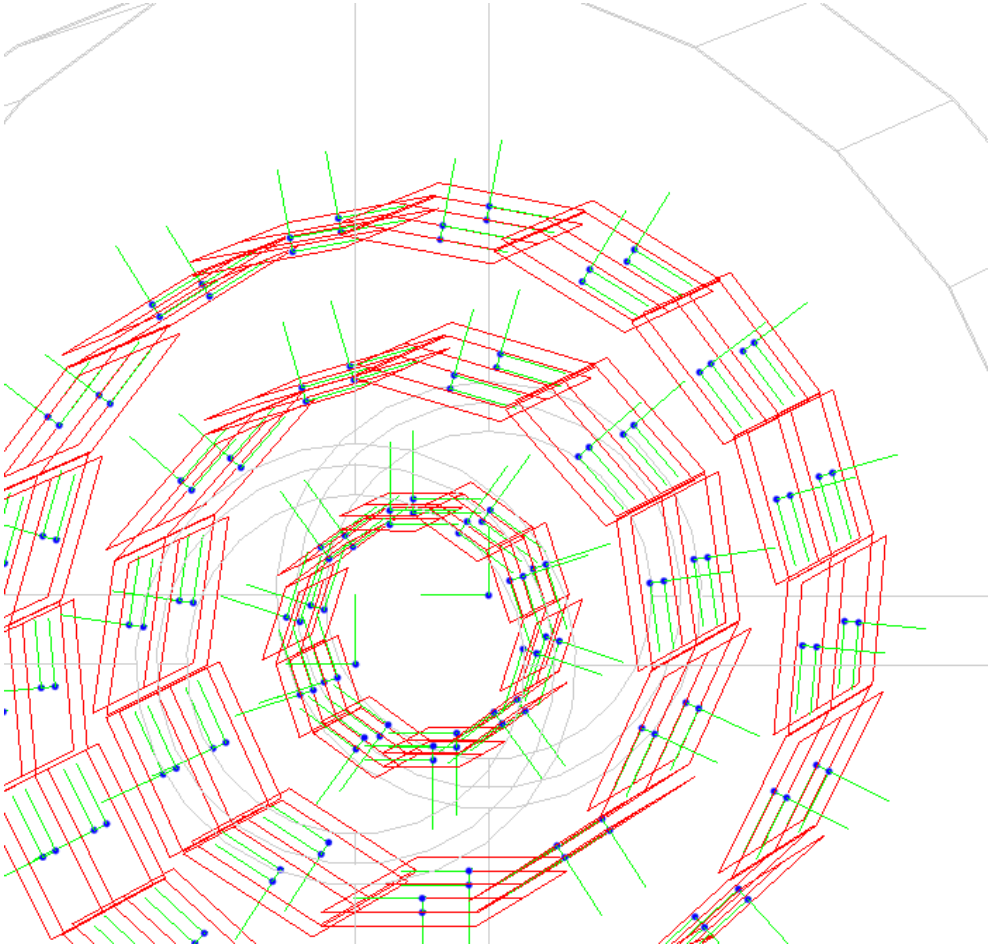
aidaTT current implementation

- IBField
  - constant B field
- IFittingAlgorithm
  - General Broken Lines (GBL)
- IGeometry
  - ISurface, IMaterial, Vector3D
  - implementation from dd4hep::DDSurfaces
    - coordinates, measurement directions, normal, material, insideBounds,...
    - tracking provides intersection with surfaces
- IPropagation
  - Analytical Propagation (perfect helix in homogeneous B field)
  - Simplified Propagation (quadratic in arc length)



DDRec surfaces for tracking

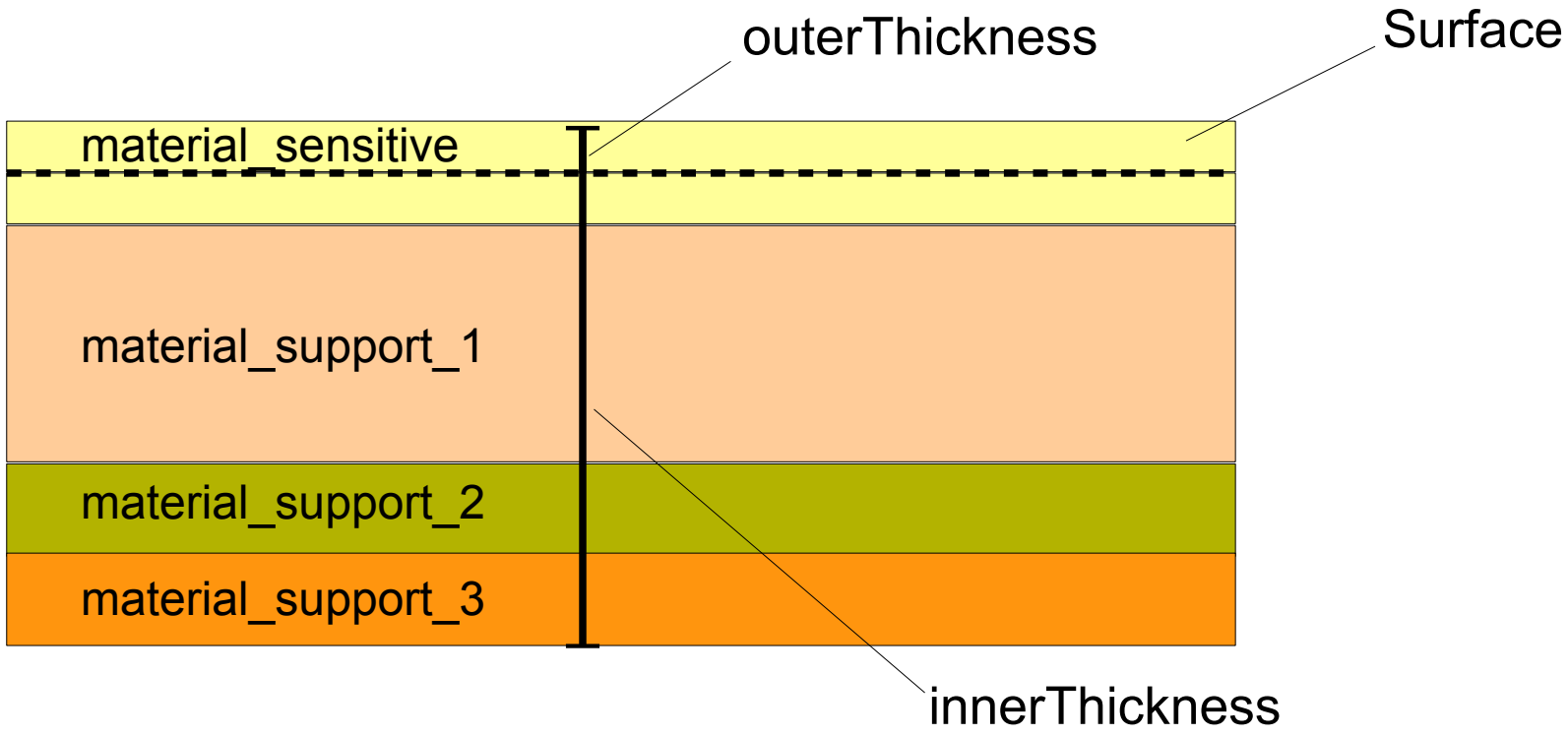
- tracking code needs a special interface to geometry:
- measurement and dead material surfaces (planar, cylindrical)
- surfaces are attached to volumes (defining boundaries) and provide:
  - u,v, normal, origin
  - inner and outer thicknesses and material
    - material is automatically averaged from detailed model
  - global to local and local to global coordinate transforms:
    - (u,v) ↔ (x,y,z)



example: surfaces attached to ILD vertex detector in new DD4hep model ILD ported from Mokka

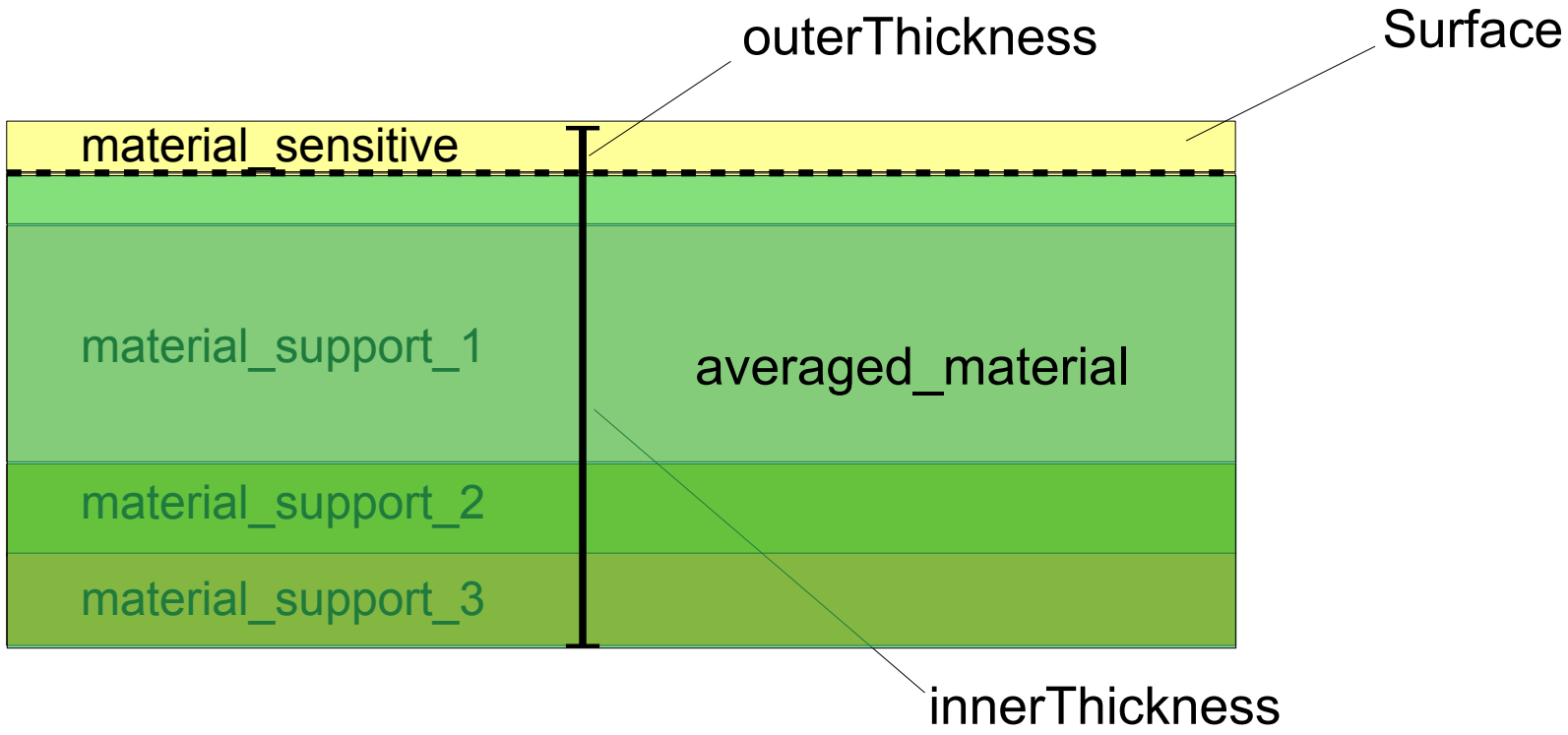
DDRec: materials and surfaces

example: Si-waver for tracking in simulation model



DDRec: materials and surfaces

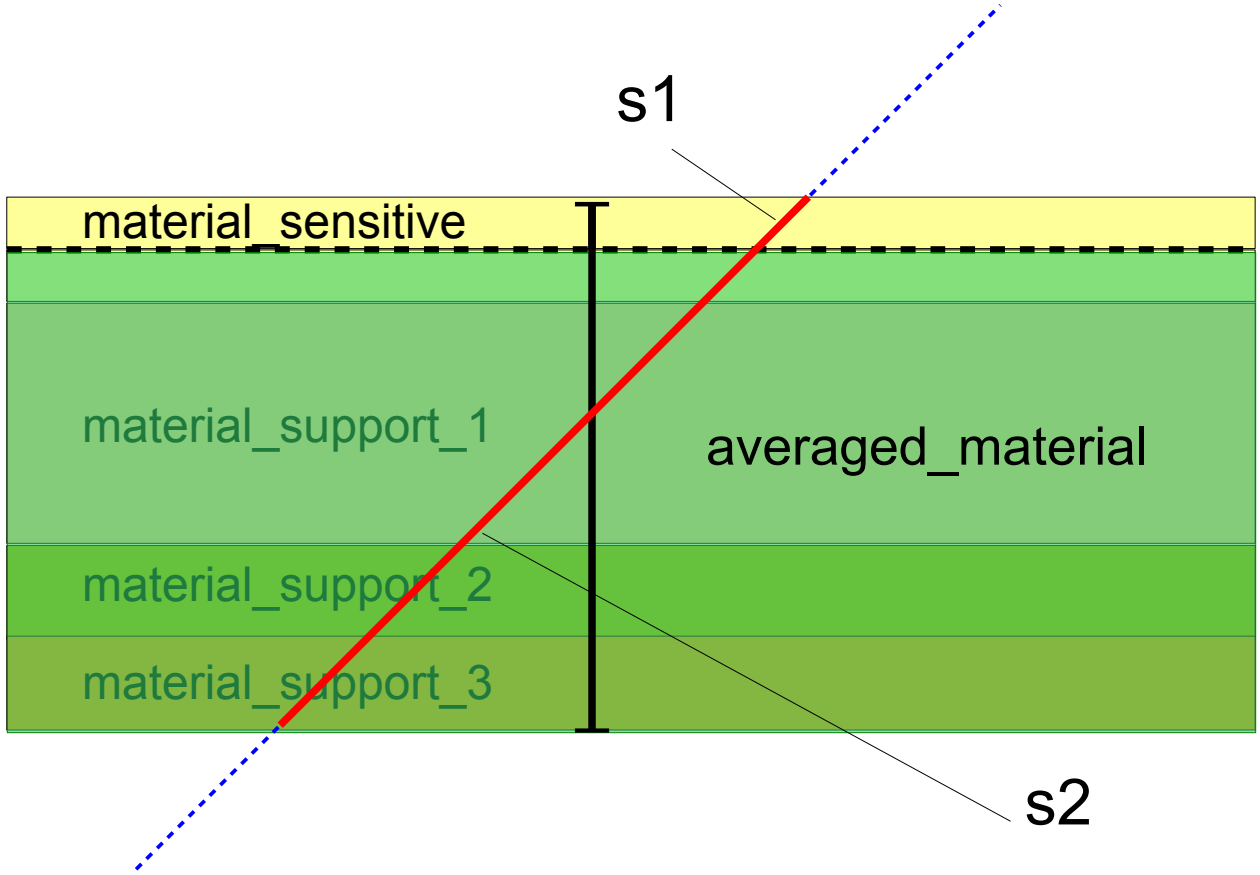
example: Si-waver for tracking in simulation model



averaged material is automatically computed from detailed simulation model

DDRec: materials and surfaces

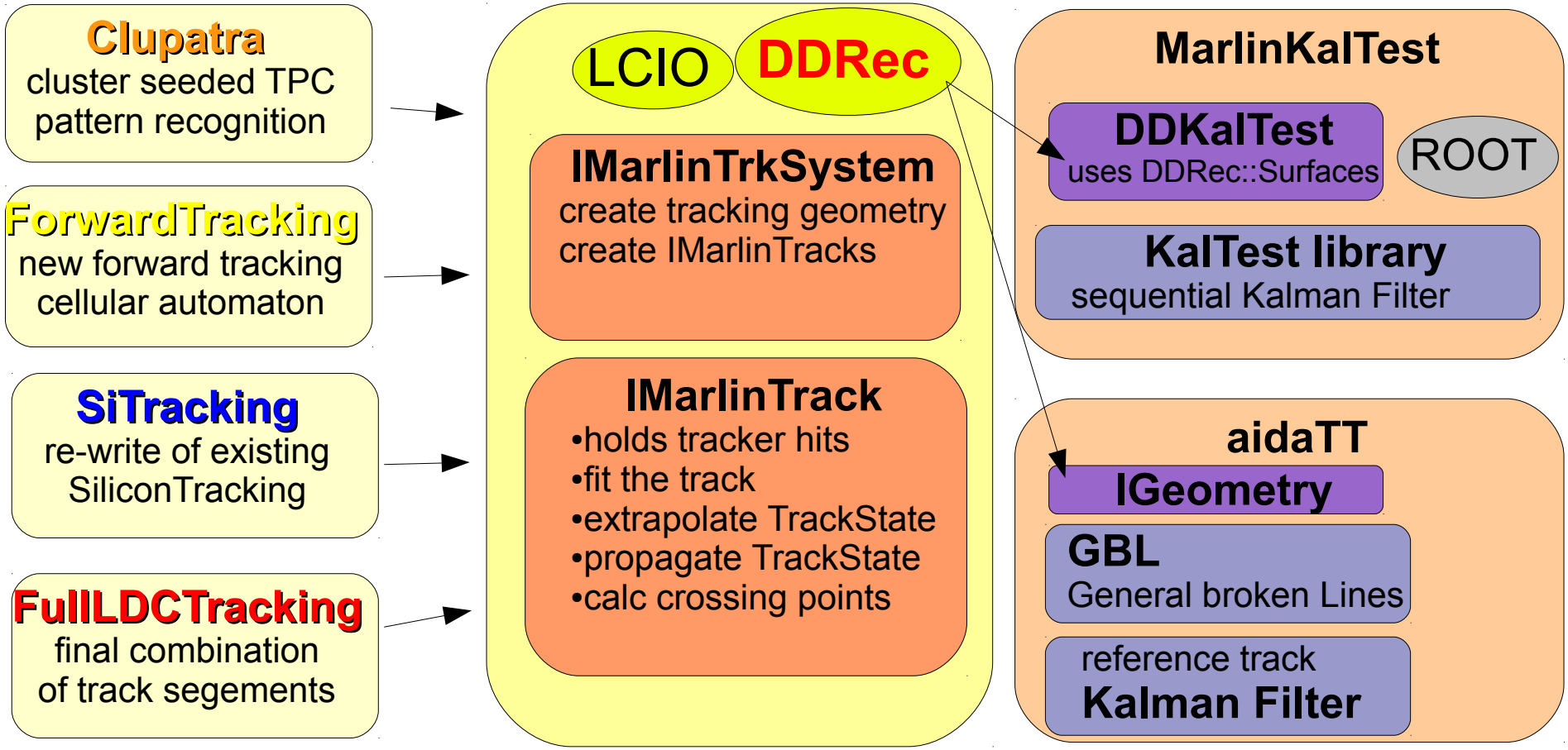
example: Si-waver for tracking in simulation model



use material properties A, Z, rho, radLen, intLen to compute effect of energy loss and multiple scattering along path lengths “through the surface” s1,s2

integration in iLCSoft tracking

- the surfaces and materials from DD4hep/DDRec will replace the pre-existing GEAR geometry description in iLCSoft
- the integration is currently ongoing



status and ongoing work

- DDKalTest
  - replaced GEAR geometry description with DDRec::Surfaces
  - implemented planar measurements for 1-d and 2-d hits
  - implemented energy loss and multiple scattering using DDRec::Material
  - to do:
    - cylindrical and disk measurement layers
    - then can run complete ILD tracking code with DD4hep based simulation
- aidaTT
  - implemented complete core functionality for track fitting with GBL
  - planar and disk measurement layers
  - interface to DDRec::Surface and DDRec::Material
  - simple example for fitting tracks from Si-Trackers DD4hep ILD model
  - to do:
    - add cylindrical layers
    - energy loss and multiple scattering
    - implement IMarlinTrk interface

Summary and Outlook

- a tracking toolkit has been developed in the context of task 2.3
- developed and tested in LC software framework
- framework independent code (packages) where possible
  - generic interface for track fitting and finding
  - pattern recognition tools based on CA and toplogical clustering
  - implementation of Kalman Filter (DDKalTest/KalTest)
  - implementation of GBL (aidaTT)
  - tracking tools use geometry description from DD4hep
    - → can be easily adapted to other detectors using DD4hep, e.g. FCC
- Outlook
  - need to add some missing functionality
  - possibly restructure software packages and write final documentation
  - provide Deliverable report D2.8 “Software Toolkit with tracking algorithms”
    - will include “High pile-up tracking tools”