Defining the VTX Geometry for LDC Simulation and Reconstruction

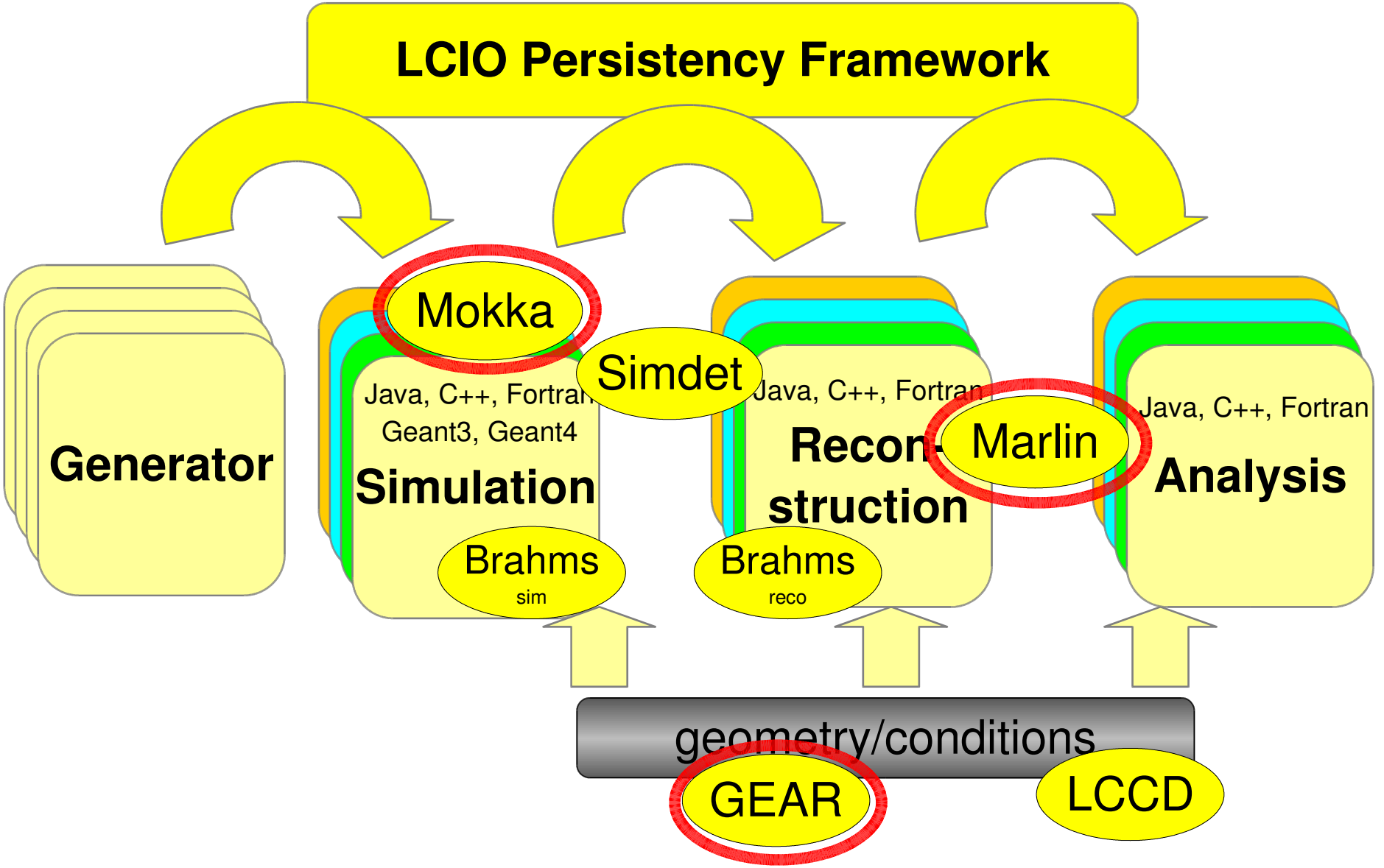
Frank Gaede
DESY

ILC Detector and Physics Workshop, Snowmass
August 14-27, 2005

Outline

- Introduction LDC simulation and reconstruction software:
  - Mokka – geant4 full simulation
  - Marlin – C++ reconstruction framework
- GEAR – geometry description
  - VTX geometry description

Overview of LDC software tools



Mokka simulation

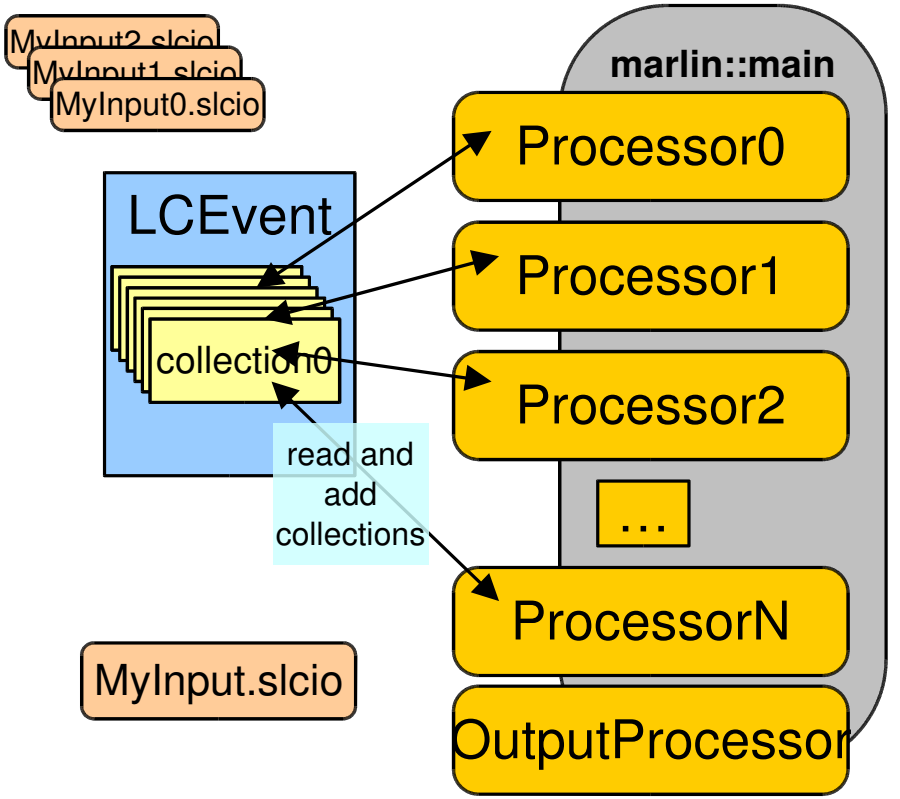
- geant4 based full detector simulation for the ILC
- developed at LLR-Ecole Polytechnique (P. Mora de Freitas, G. Musat)
- http://polywww.in2p3.fr/geant4/tesla/www/\mokka/mokka.html
- some features:
  - steering files for configuration
  - all geant4 physics lists available
  - writes LCIO
  - reads StdHep / ASCII HepEvt
- Geometry
  - MySQL databases
    - geometry parameters
    - one database per subdetector
  - C++ Geometry Drivers
    - one for each subdetector type (e.g. TPC, HCAL)
    - define material and sensitive detector
  - abstract geometry layer: CGA

new features: (v05-01)
- SID detector model
- scalable detector models

Marlin

Modular**A**nalysis & **R**econstruction for the **L I N**ear Collider

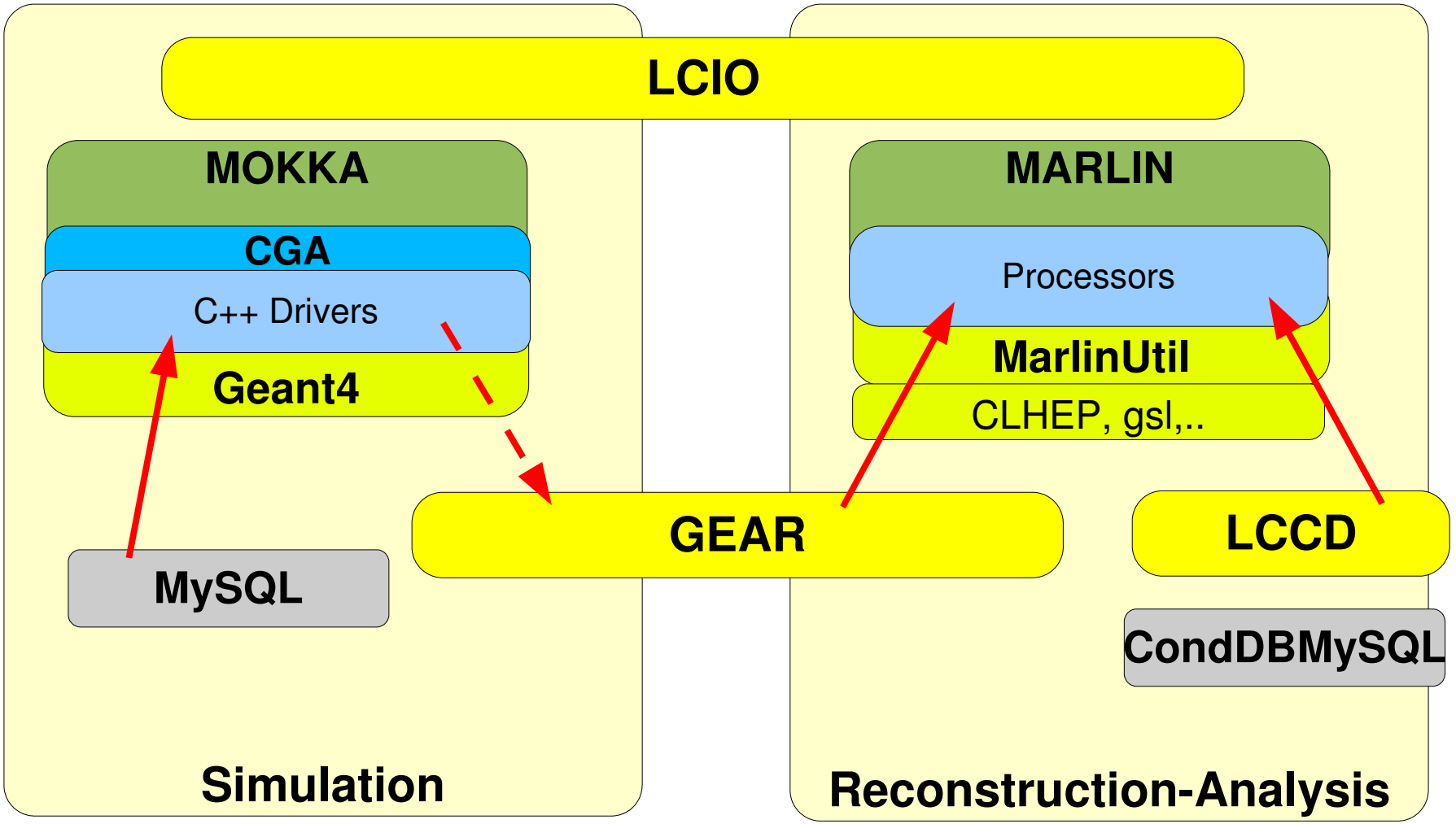
- modular C++ **application framework** for the analysis and reconstruction of LCIO data
- uses LCIO as transient data model
- software modules called Processors
- provides main program !
- provides simple user steering:
  - program flow (active processors)
  - user defined variables
    - per processor and global
  - input/output files



MarlinReco

- **Marlin serves as a framework for the distributed development of a full suite of reconstruction algorithms !**
  - **-> your input is welcome**
- **(almost) complete set of standard reconstruction in Marlin available: MarlinReco** (see talk by S. Aplin)
  - Clustering in Ecal and Hcal
  - Tracking in TPC only so far
- uses first implementation of GEAR geometry description

LDC simulation framework



GEAR Overview

GEometry **A**PI for **R**econstruction

- Motivation:
  - need well defined geometry definition that
    - is flexible w.r.t different detector concepts
    - has high level information needed for reconstruction
      (different from detailed local description for simulation !)
    - supports 'plug & play' philosophy of processors implementing different algorithms
    - provides access to material properties (radiation/interaction lengths)
- Idea:
  - define abstract interface (a la LCIO C++ and Java ?)
  - concrete implementation based on XML files and CGA

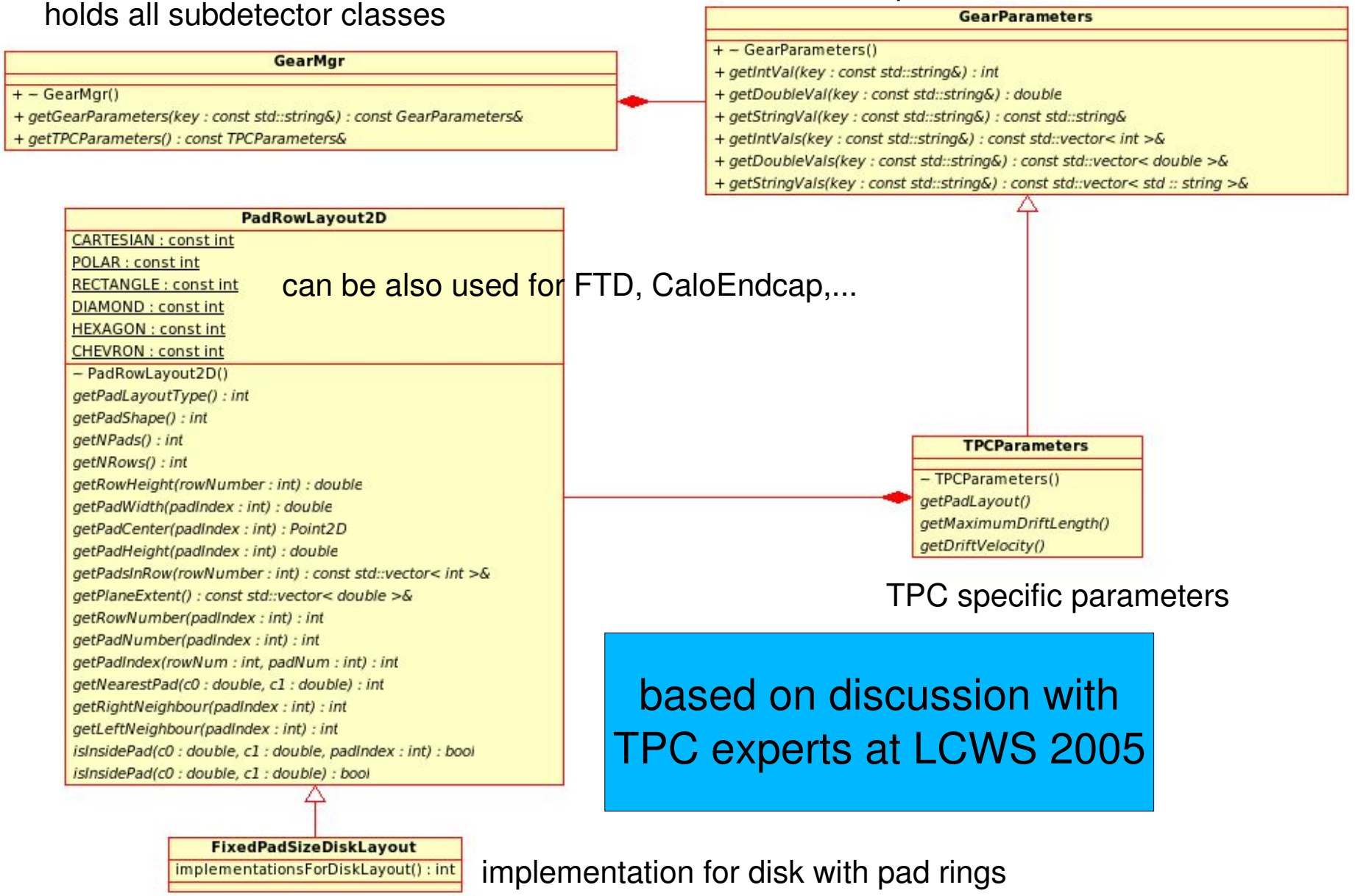
GEAR – Classes

- Subdetector description
  - high level description of detector shape and readout geometry – one class for every subdetector type, e.g.
    - TPC, Ecal, Hcal (MainCalorimeter), FTD, VTX, SIT, ...
    - defines required attributes - as detailed as necessary but as abstract as possible
    - allows to add additional named attributes
  - use XML files
- Material properties
  - point properties (density, material, radlen,...)
  - distance properties integrated along (straight!?) path
  - use Mokka-CGA interface to geant4 geometry

GEAR – TPC description

holds all subdetector classes

named parameters for additional attributes



can be also used for FTD, CaloEndcap,...

TPC specific parameters

based on discussion with TPC experts at LCWS 2005

implementation for disk with pad rings

GEAR – material properties

GearDistanceProperties

- GearDistanceProperties()
getMaterialNames(p0 : const Point3D&, p1 : const Point3D&) : const std::vector< std :: string >&
getMaterialThicknesses(p0 : const Point3D&, p1 : const Point3D&) : const std::vector< double >&
getNRadlen(p0 : const Point3D&, p1 : const Point3D&) : double
getNIntlen(p0 : const Point3D&, p1 : const Point3D&) : double
getBdL(pos : const Point3D&) : double
getEdL(pos : const Point3D&) : double

integrated along path:
- straight line or
- true path in B-Field ?

GearPointProperties

- GearPointProperties()
getCellID(pos : const Point3D&) : int
getMaterialName(pos : const Point3D&) : const std::string&
getDensity(pos : const Point3D&) : double
getTemperature(pos : const Point3D&) : double
getPressure(pos : const Point3D&) : double
getRadlen(pos : const Point3D&) : double
getIntlen(pos : const Point3D&) : double
getLocalPosition(pos : const Point3D&) : Point3D
getB(pos : const Point3D&) : double
getE(pos : const Point3D&) : double
getListOfLogicalVolumes(pos : const Point3D&) : std::vector< std :: string >
getListOfPhysicalVolumes(pos : const Point3D&) : std::vector< std :: string >
getRegion(pos : const Point3D&) : std::string
isTracker(pos : const Point3D&) : bool
isCalorimeter(pos : const Point3D&) : bool

properties at point from geant4 (CGA)

based on discussions at Argonne Simulation Meeting 2004

GEAR status and outlook

- first mini implementation with XML files for current MarlinReco: TPC, Ecal, Hcal
  - XML format 'compatible' with US compact description
  - integrated in Marlin v00-09-01 (see DVD)
  - soon to be released !
- Plans:
  - have discussions at snowmass with other subdetector groups about abstract interface needed
  - -> now with VTX group !
  - provide implementation of material properties based on CGA/Mokka
  - investigate option of creating GEAR XML files in Mokka geometry drivers

describing the VTX geometry

- Mokka full simulation VTX-FTD :
  - so far: simple cylinders and disks only
  - -> do we need sth. more realistic with polygons or staggered ladders ?
  - is someone willing to code sth. else ?
- need to find abstract description of VTX detector for GEAR
  - what level of detail is needed for track reconstruction ?
  - need input from people working on VTX tracking !

start discussion now !