Introduction and Overview ILC software

Frank Gaede
DESY

EUDET – Annual Meeting,
MPI Munich
October 18-20, 2006

Session: Software & Computing

- Wednesday Oct 18, 14:00-18:00 h
- Introduction and Overview ILC software
  - F.Gaede (20 min)
- Status JRA2 Software
  - P.Wienemann (20 min)
- Grid computing from a users' point of view
  - A.Vogel (20 min)
- Discussion and Plans: common software
  - all (~ 30 min)
- Status JRA3 Software
  - R.Poeschl (-> tomorrow DAQ session @ 16:00 h)

Outline

- Task NA2 – ANALYS
  - ILC software in the context of EUDET
  - Core software tools
    - status and latest developments
      - LCIO
      - Marlin
      - LCCD
      - GEAR
  - A new geometry package: LCGO (proposal)
  - Summary

Objectives for task ANALYS

- **development of a common data analysis and simulation infrastructure**
  - development of a software framework using modern software technology to exchange test beam data and software for common analysis and comparison of measurements
  - development of a software framework for the simulation of test beam experiments needed for the interpretation of the measurements
  - creation of a repository for experimental and simulation data
  - embedding into existing GRID infrastructure to allow easy exchange of data and transparent exploitation of other available computing resources.

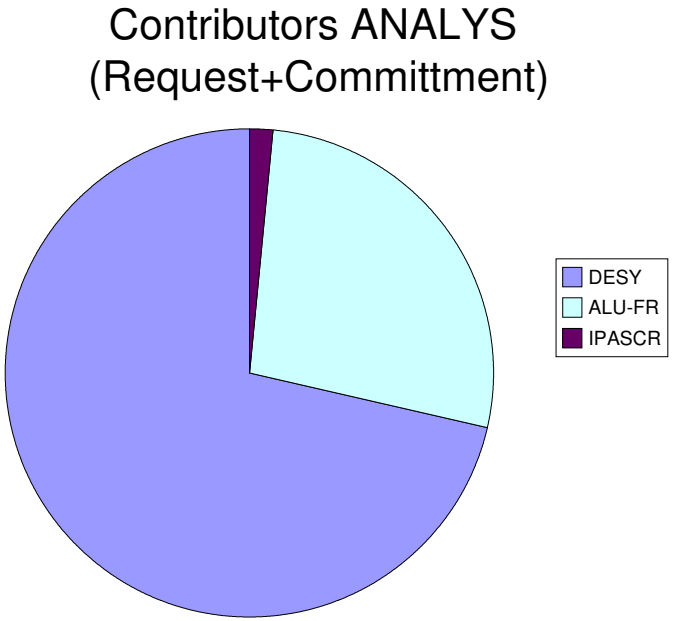
(from annex1)

Usage of budget - ANALYS

- DESY
  - commitment 12ppm: F.Gaede 25% for full project length
  - 12ppm (scientist) converted to hire a programmer for 18 month
    - **started August 2006**
    - (possibly extend position with other funding sources)

- RFWU-Bonn (ALU-FR) (K.Desch, P. Wienemann)
  - request: 8ppm (scientist): plan to combine with funds (8ppm) from COMP to hire a postdoc that works part-time on COMP and ANALYS
    - will start in October (Nov.) 2006

- IPASCR (J.Cvach)
  - commitment: 3ppm: PhD student that works
  - part time on calorimeter simulation with geant4
    - not yet



General strategy for ANALYS

- there will be no EUDET/testbeam specific simulation and analysis software framework !
  - avoiding of double work
  - a lot of what's needed already exists
- the testbeam software effort is tightly integrated with the overall common ILC/LDC software effort !
  - implement tools and functionality specific to testbeams
  - benefit from synergies where possible, e.g. use geant4 application for full detector also for testbeam (Mokka/Calice)
- same for grid tasks: integrate with common ILC grid activities

Software packages for ILC framework

- LCIO
  - data model & persistency
  - (Mokka)
    - geant4 full simulation
- Marlin
  - C++ application framework
  - (MarlinReco)
    - Marlin based reconstruction
- LCCD
  - conditions data toolkit
- GEAR
  - geometry description

all packages developed at or with contributions from DESY

-> DESY will naturally continue to develop and improve these tools in the context of EUDET

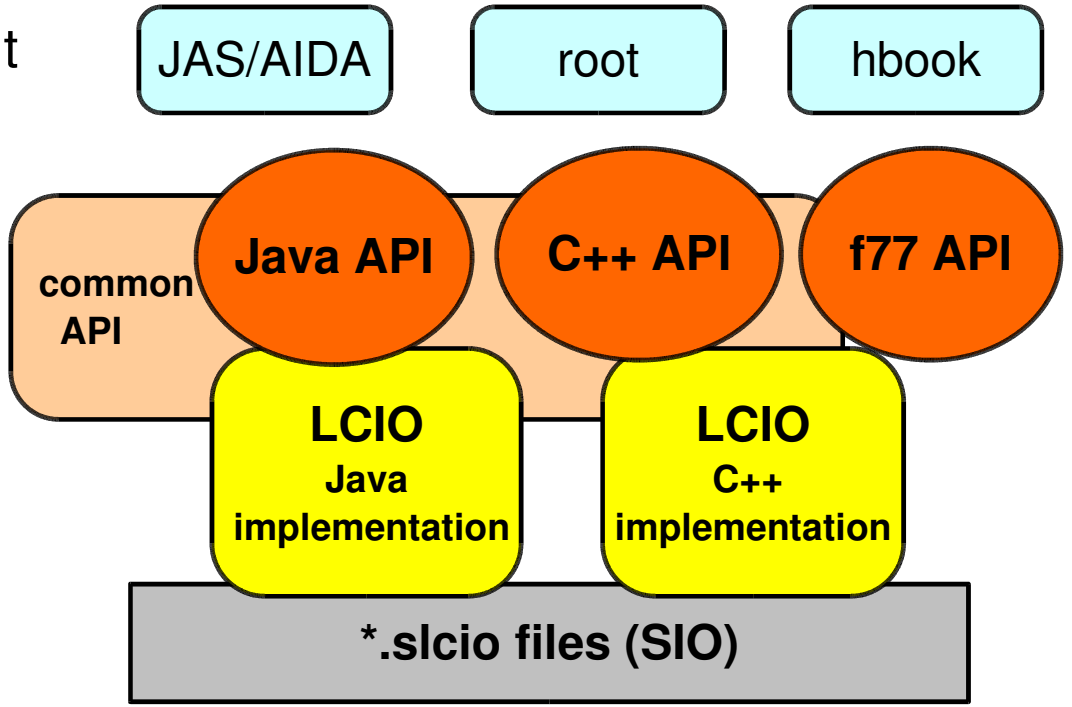
LCIO overview

- DESY and SLAC joined project:
  - provide common basis for ILC software
- Features:
  - Java, C++ and f77 (!) API
  - extensible data model for current and future simulation and testbeam studies
  - user code separated from concrete data format
  - no dependency on other frameworks

simple & lightweight

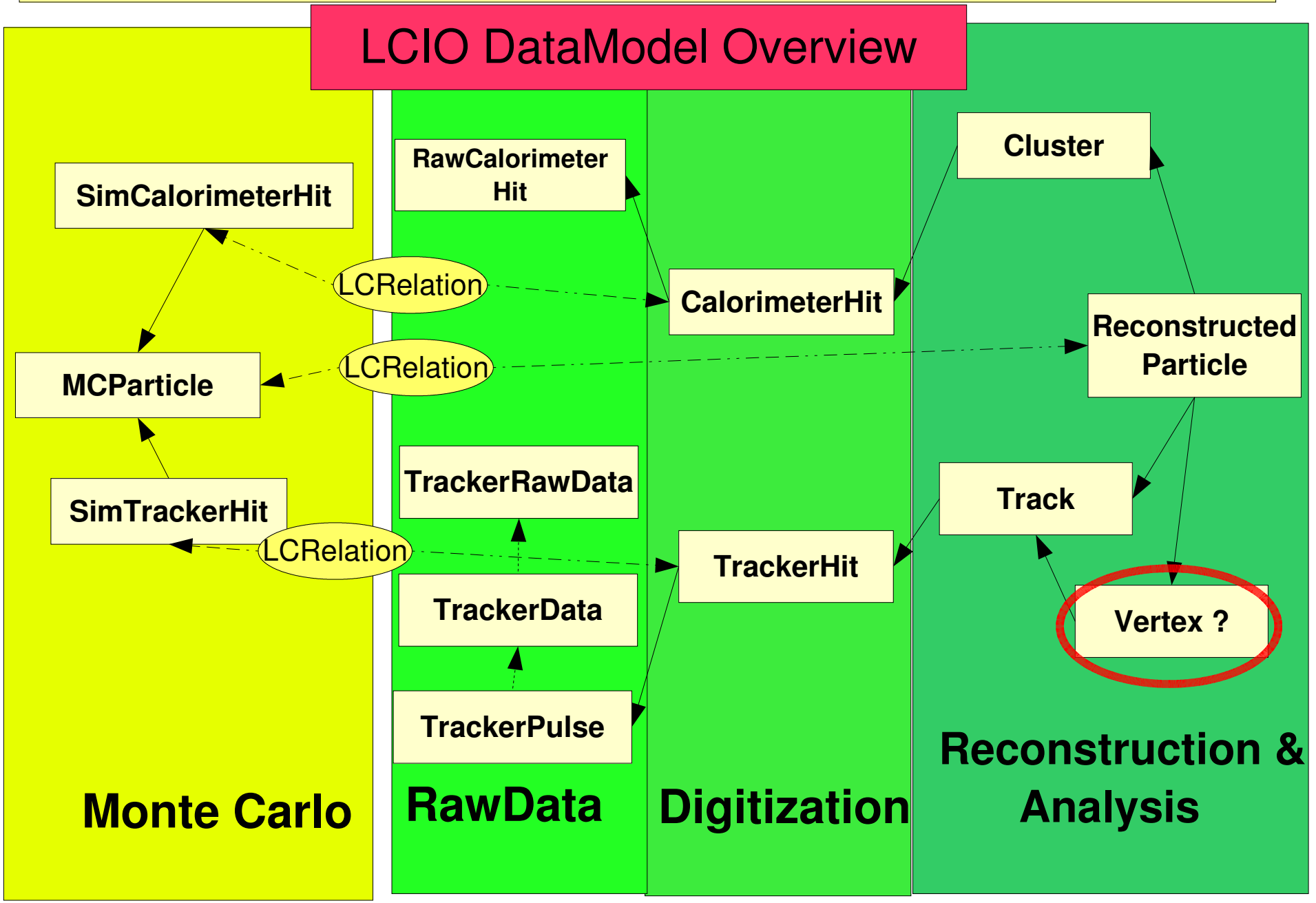
now de facto standard persistency & datamodel for ILC software

SW-Architecture



current release: **v01-07**

LCIO Event Data Model I



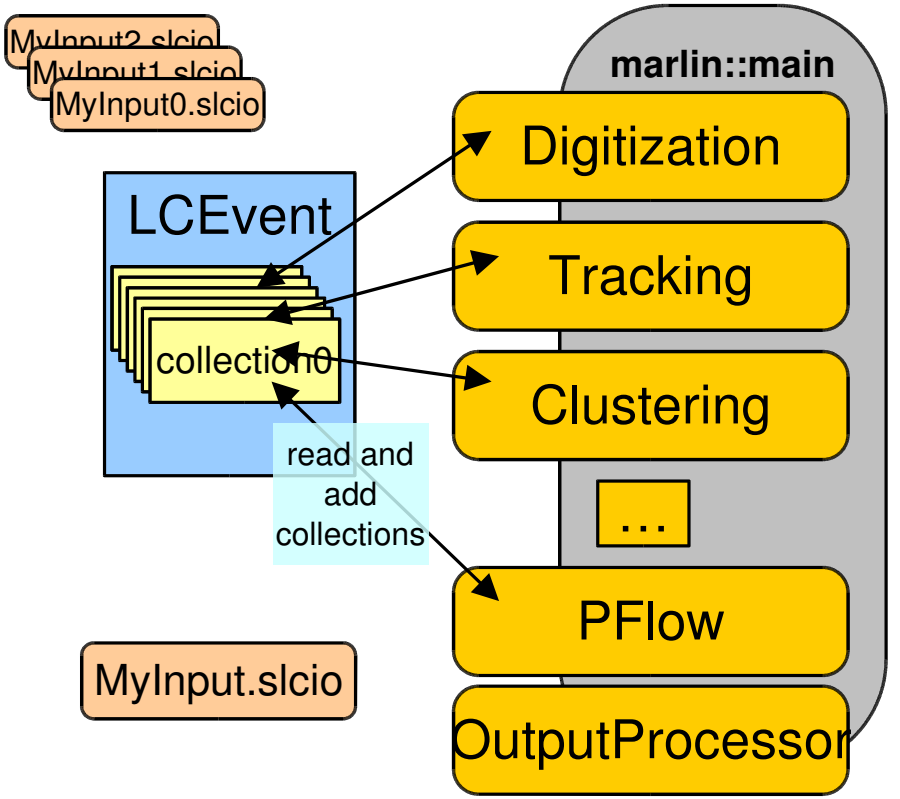
LCIO Event Data Model II

- the LCIO event data model is fairly complete and flexible
- however it is adapted and extended as needed by the community
  - maintaining downward compatibility
  - with international discussion and agreement
  - example: introduction of a new **Vertex** class in LCIO
    - originally proposed by LCFI group
      - see discussion @ http://forum.linearcollider.org/
    - test release v01-08-vtx
  - new raw data classes for prototypes
    - TPC uses TrackerRawData, TrackerData, TrackerPulse
      - also to be used for vertex prototypes
    - calorimeter (calice) could/should use RawCalorimeterHit or extension/additional classes

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
  - **Plug&Play** of processors



Marlin – XML steering files

```
- <marlin>
  - <execute>
      <processor name="MyAIDAProcessor"/>
      <processor name="MyEventSelection"/>
    - <if condition="MyEventSelection">
        <group name="Tracking"/>
        <processor name="MyClustering"/>
        <processor name="MyPFlow"/>
        <processor name="MyLCIOOutputProcessor"/>
      </if>
    </execute>
  - <global>
      <parameter name="LCIOInputFiles"> simjob.slcio </parameter>
      <parameter name="MaxRecordNumber" value="5001"/>
      <parameter name="SupressCheck" value="false"/>
    </global>
  - <processor name="MyLCIOOutputProcessor" type="LCIOOutputProcessor">
      <parameter name="LCIOOutputFile" type="string">outputfile.slcio </parameter>
      <parameter name="LCIOWriteMode" type="string">WRITE_NEW</parameter>
    </processor>
  - <group name="Tracking">
      <parameter name="NTPCLayers" value="200"/>
      <processor name="MyTrackfinder" type="Trackfinder"/>
    - <processor name="MyTrackfitter" type="Trackfitter">
        <parameter name="Algorithm" value="DAF"/>
      </processor>
    </group>
    <!-- ... -->
  </marlin>
```

- Program flow defined in <execute>...</execute> section
- logical conditions from parameters evaluated at runtime

- global Parameters defined in <global/> section

- local Parameters defined in mandatory <parameter/> section

- Processors can be enclosed by <group/> tag
- Parameters in <group/> joined by all processors

a Marlin application is fully configured through the steering files (no user main program) !!

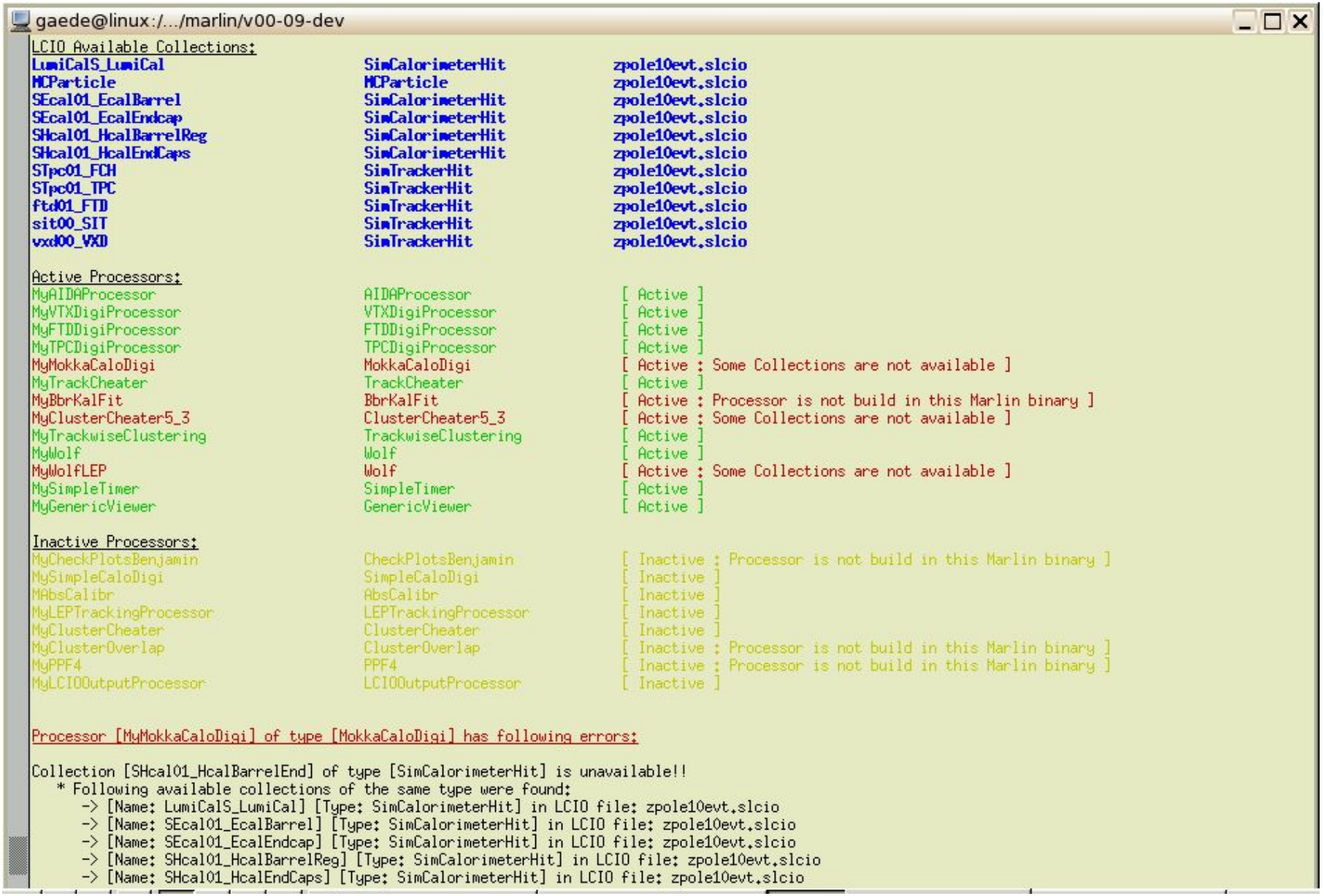
Marlin Status and Plans

- current version: v00-09-05:
  - made compatible with CLHEP 2.x
  - made compatible with RAIDA (root impl. of AIDA)
  - split outputfiles wrt. size: ttbar_000.slcio, ttbar_001.slcio,...
- new version (soon)

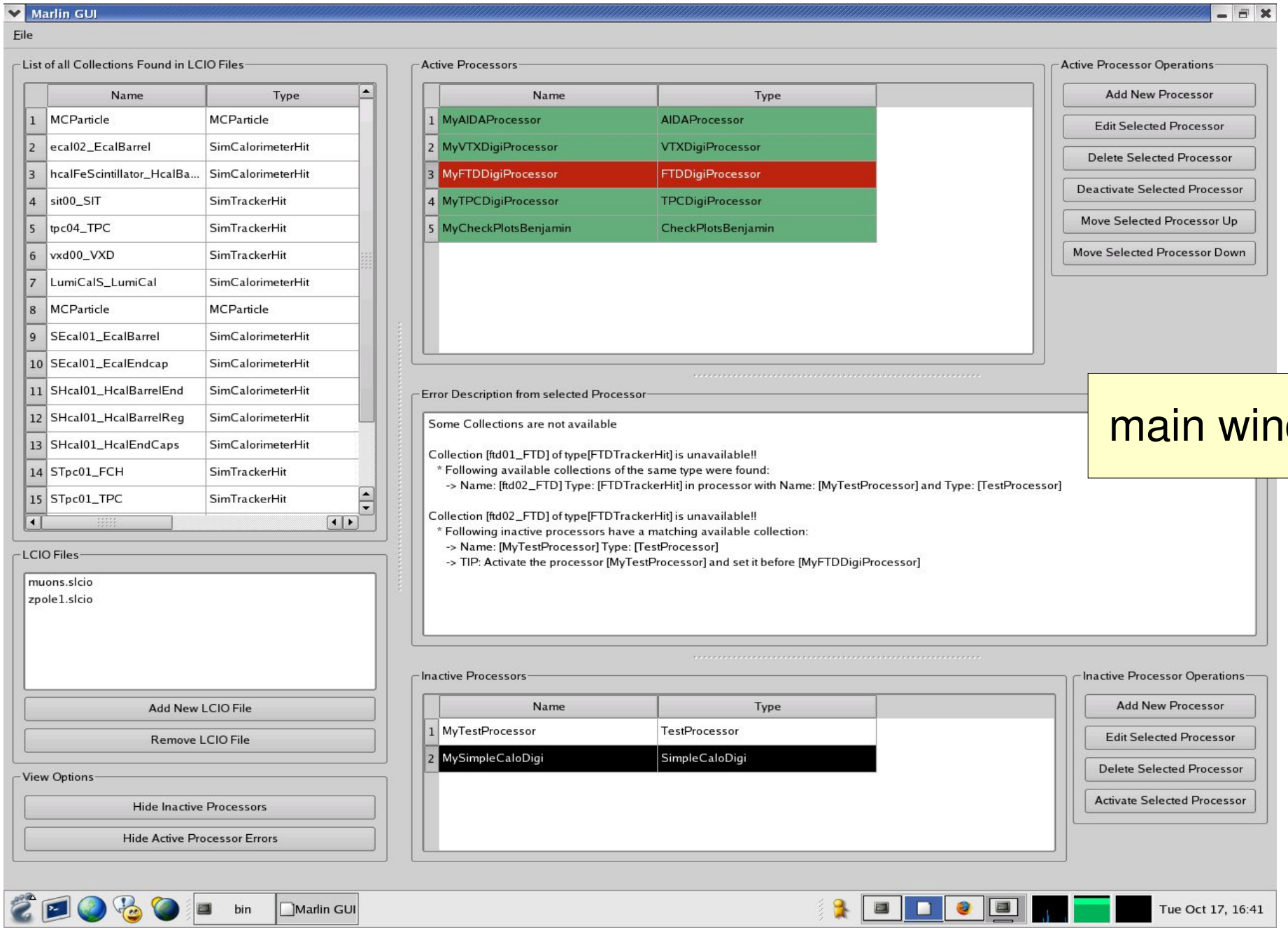
J.Engels (EUDET)

  - Marlin -c mysteer.xml
    - consistency check and prints error messages/hints
  - Marlin -f oldsteer.xml newsteer.xml
    - adds LCIO collection type information
  - Processor::addInput/OutputCollection() to be called by users/authors of marlin processors
  - MarlinGUI:
    - interactive creation/modification of steering files

Example: Marlin -c steer.xml

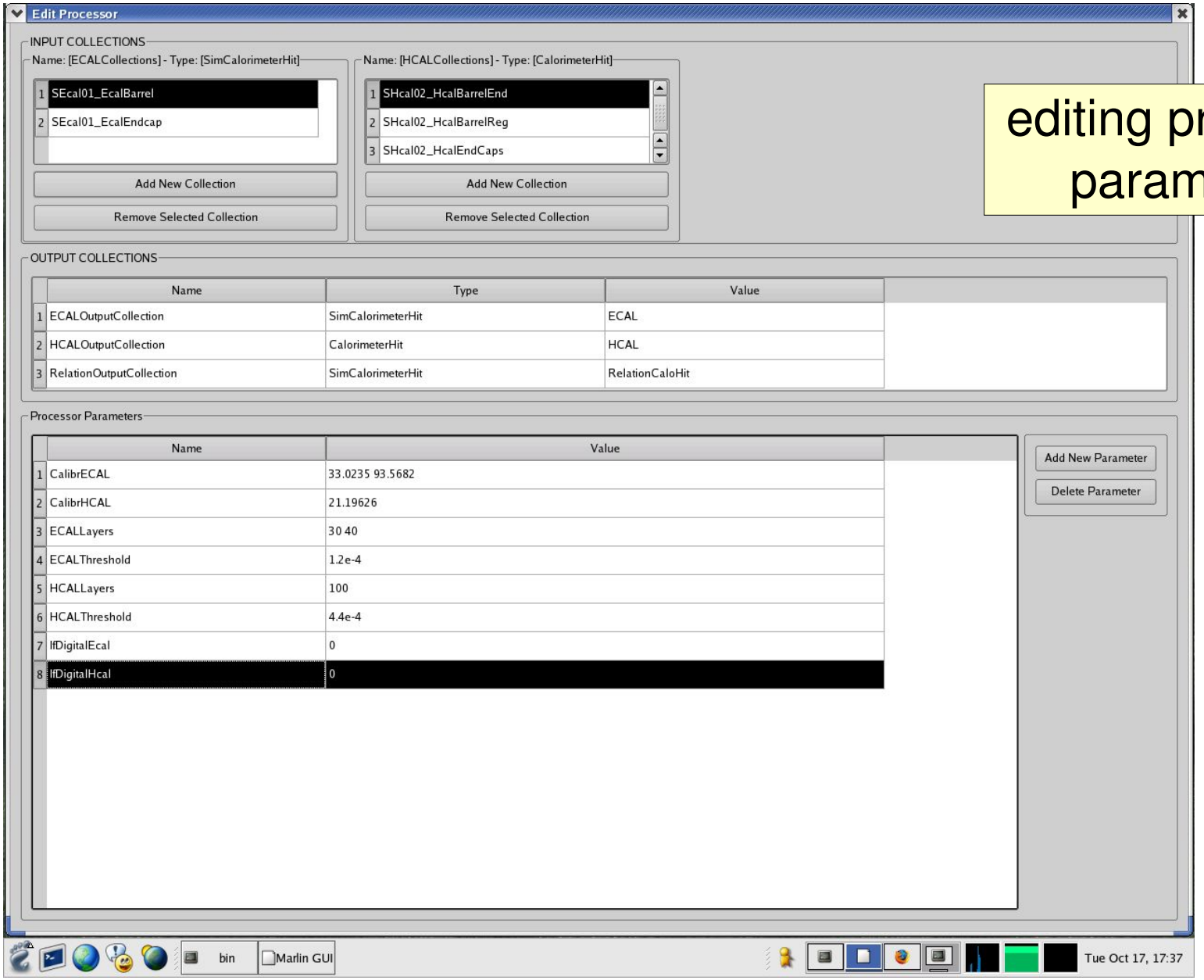


example: MarlinGUI I



main window

example: MarlinGUI II

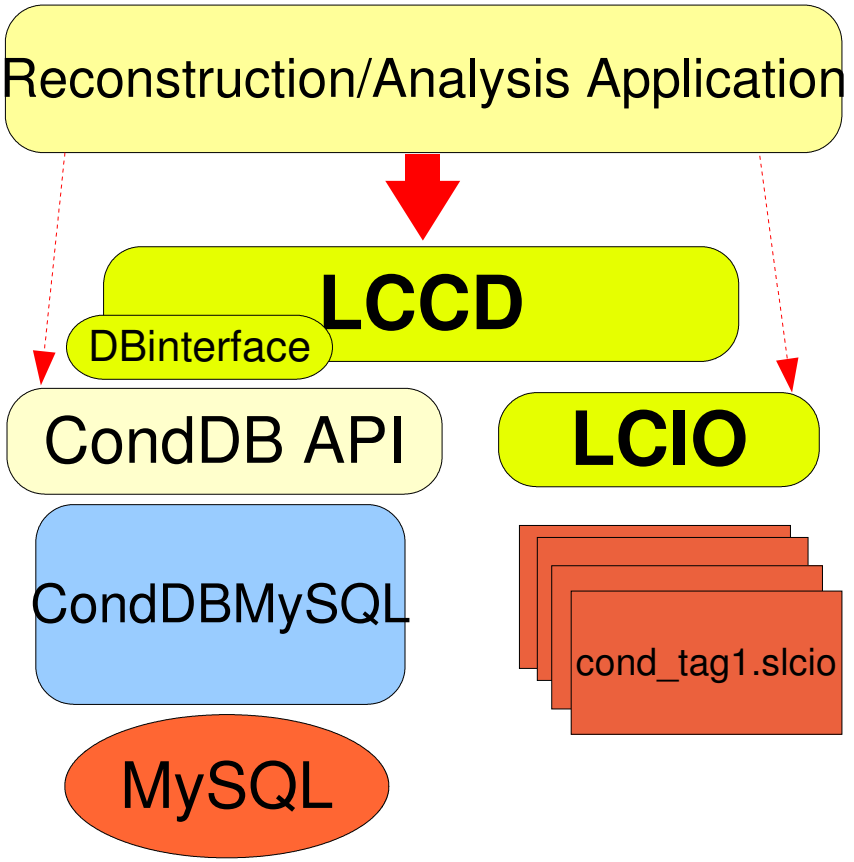


editing processor parameters

LCCD

Linear **C**ollider **C**onditions **D**ata Toolkit

- Reading conditions data
  - from conditions database
  - from simple LCIO file
  - from LCIO data stream
  - from dedicated LCIO-DB file
- Writing conditions data
- tag conditions data
- Browse the conditions database
  - through creation of LCIO files
    - vertically (all versions for timestamp)
    - horizontally (all versions for tag)



LCCD is used by Calice for the conditions data of the ongoing testbeam studies

Gear

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

```
- <gear>
  - <!--
      Example XML file for GEAR describing the LDC detector
    -->
  - <detectors>
    - <detector id="0" name="TPCTest" geartype="TPCParameters" type
        <maxDriftLength value="2500."/>
        <driftVelocity value=""/>
        <readoutFrequency value="10"/>
        <PadRowLayout2D type="FixedPadSizeDiskLayout" rMin="386.0"
        maxRow="200" padGap="0.0"/>
        <parameter name="tpcRPhiResMax" type="double"> 0.16 </para
        <parameter name="tpcZRes" type="double"> 1.0 </parameter>
        <parameter name="tpcPixRP" type="double"> 1.0 </parameter>
        <parameter name="tpcPixZ" type="double"> 1.4 </parameter>
        <parameter name="tpcIonPotential" type="double"> 0.00000003
      </detector>
    - <detector name="EcalBarrel" geartype="CalorimeterParameters">
        <layout type="Barrel" symmetry="8" phi0="0.0"/>
        <dimensions inner_r="1698.85" outer_z="2750.0"/>
        <layer repeat="30" thickness="3.9" absorberThickness="2.5"/>
        <layer repeat="10" thickness="6.7" absorberThickness="5.3"/>
      </detector>
    - <detector name="EcalEndcap" geartype="CalorimeterParameters">
        <layout type="Endcap" symmetry="2" phi0="0.0"/>
        <dimensions inner_r="320.0" outer_r="1882.85" inner_z="2820.
        <layer repeat="30" thickness="3.9" absorberThickness="2.5"/>
        <layer repeat="10" thickness="6.7" absorberThickness="5.3"/>
      </detector>
    </detectors>
  </gear>
```

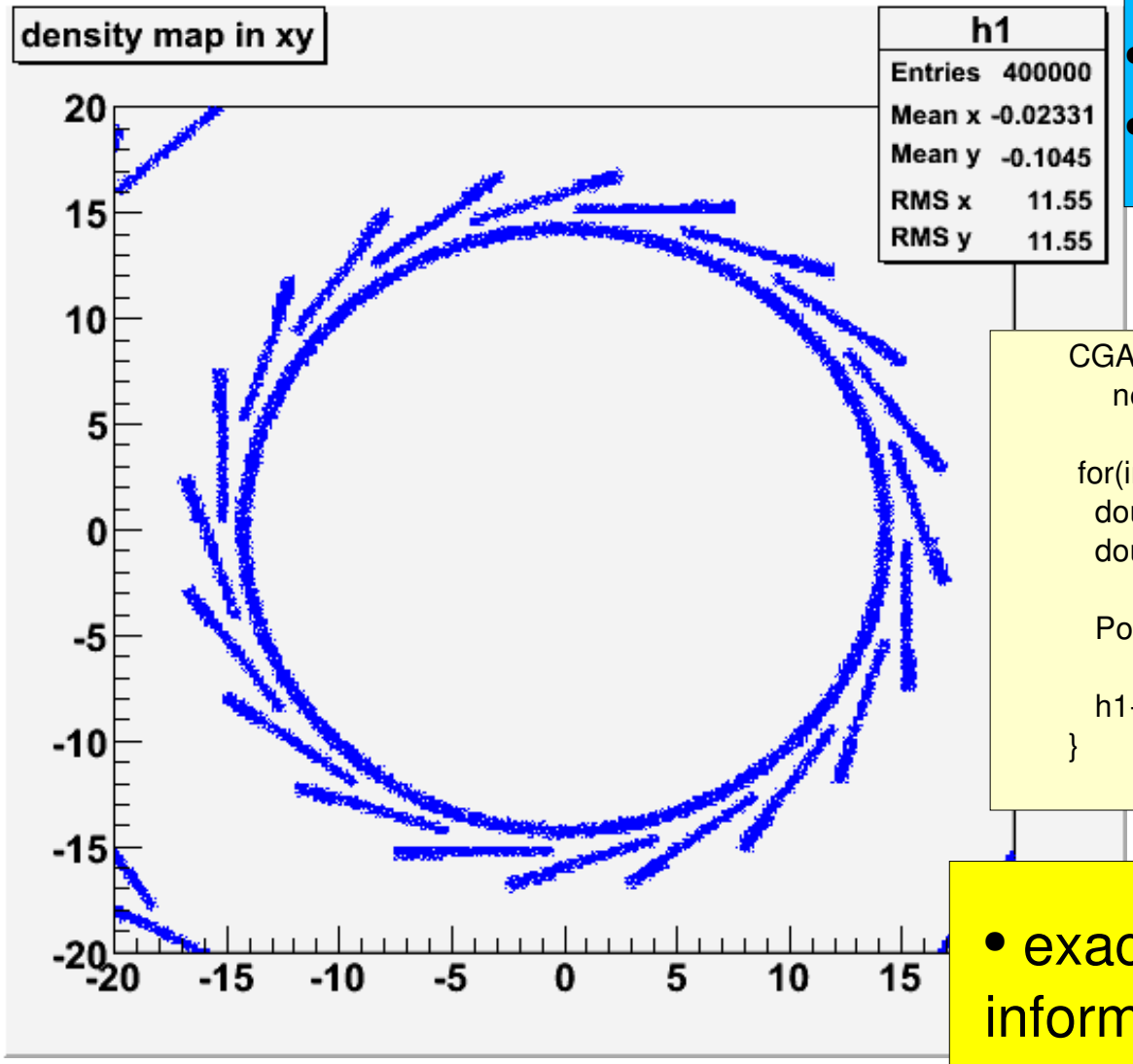
compatible with US – compact format

- well defined geometry definition for reconstruction that
  - is flexible w.r.t different detector concepts
  - has high level information needed for reconstruction
  - provides access to material properties
- abstract interface (a la LCIO)
  - concrete implementation based on XML files
  - and Mokka-CGA

Gear status

- version v00-03
  - TPC, Hcal, Ecal and VXD (new) interfaces defined and implemented
  - user parameters
  - write xml files from parameters in memory
  - tool to merge files: gearmerge
  - description of TPC prototypes (rectangular pad plane)\
- GearCGA - material properties
- related work: MokkaGear

CGAGear



- implemented by G.Musat, LLR
- to be released soon

```
CGAGearPointProperties * pointProp =
    new CGAGearPointProperties(steer.str(),...);

 for(int i=0 ; i<nPoint ; ++i){
   double xr =  xmin +  ( xmax - xmin)  * random() ;
   double yr =  ymin +  ( ymax - ymin)  * random() ;

   Point3D p( xr, yr, z0 ) ;

   h1->fill(  xr, yr, pointProp->getDensity( p )  ) ;
}
```

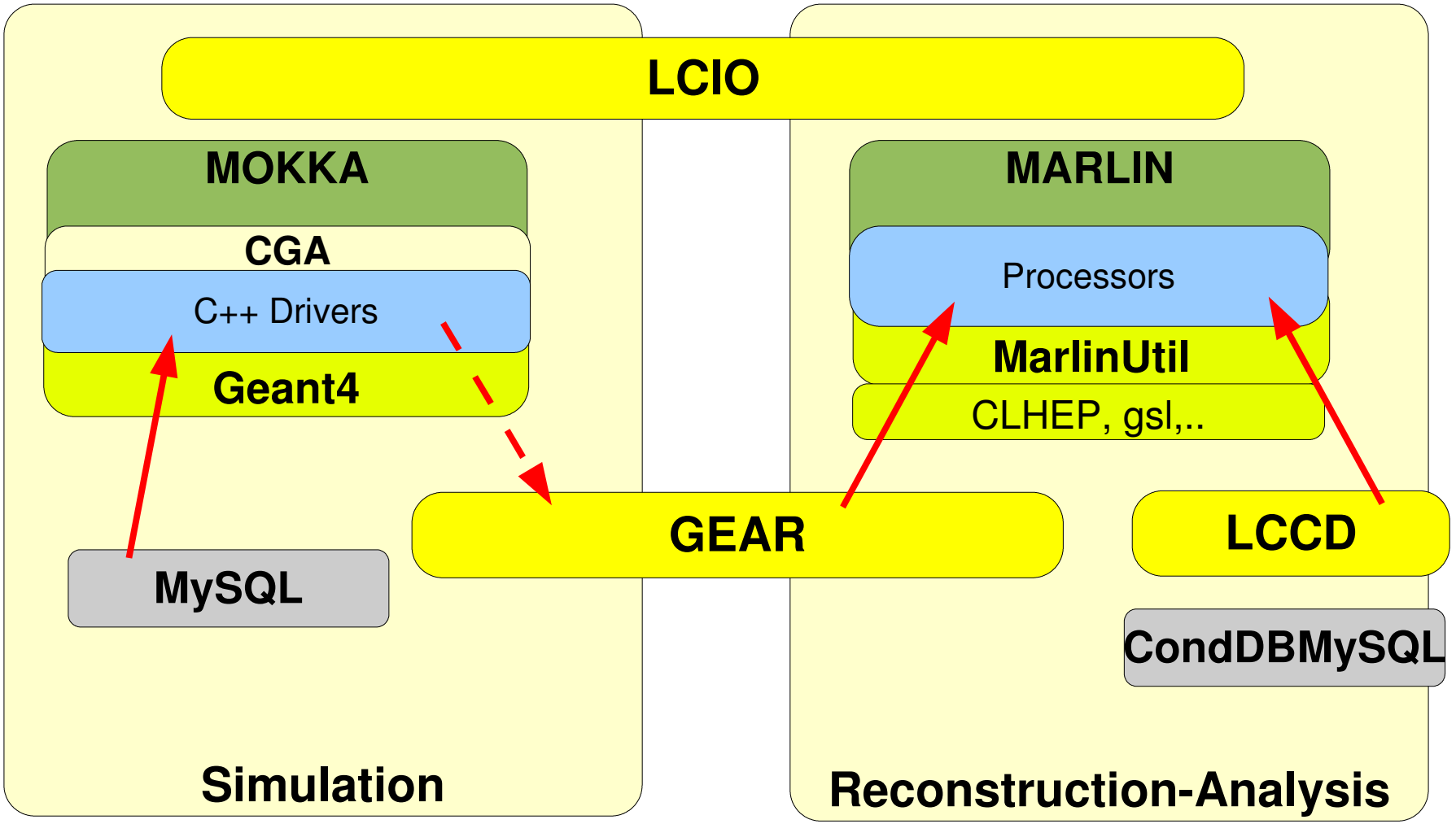
- exact geant4 material & field information at runtime !
- performance ?
- practical issues (linking g4) ?

MokkaGear

- extension to Mokka (R.Lippe – Diploma Thesis)
- extract geometry information in drivers when detector is built
- use Gear to create XML files for reconstruction
- currently implemented:
  - TPC (tpc04), Ecal (ecal02) and Hcal (hcal04)
- **released with Mokka 6.1**
- optional feature
  - only if Gear is installed and included

aim: have only one source of of information
for describing the detector geometry !

LDC software framework



all tools are also used in testbeam programs

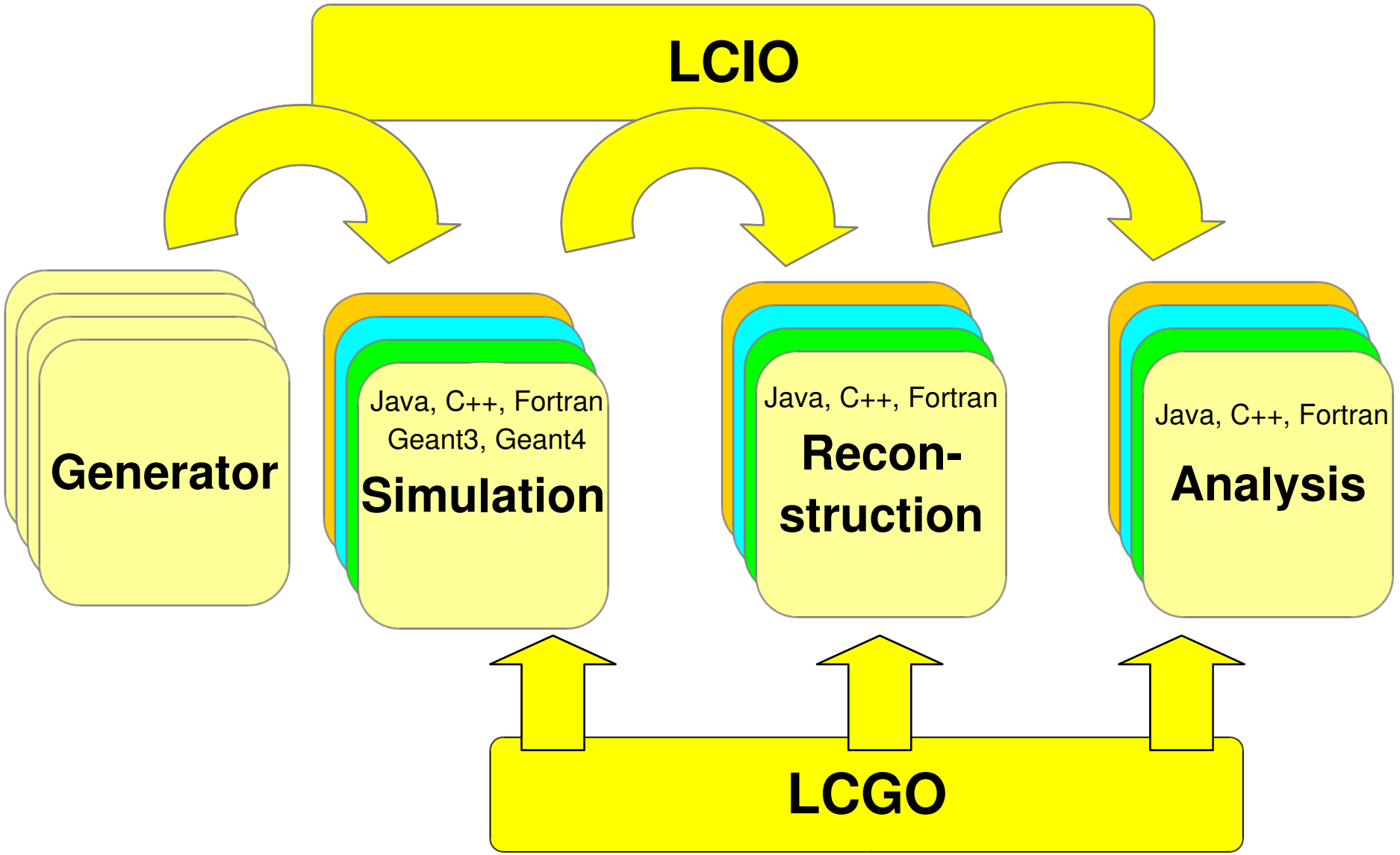
ILC software used in testbeams

- CALICE PPT-testbeam (now at CERN)
  - usage of LCIO, Marlin, Gear, LCCD
  - specific extensions developed by CALICE
- TPC prototypes
  - usage of LCIO, Marlin, Gear, (LCCD planned)
  - special raw data classes for Tracker hits in LCIO
  - Gear geometry description of TPC prototype
- VTX prototypes
  - usage of LCIO, Marlin, Gear
  - development of VTX geometry definition in Gear

A Common Geometry Toolkit

- **LCGO**: A common geometry toolkit to be used in all (?) ILC frameworks
  - SLAC-DESY project - initially
  - -> of course open for all collaborators, e.g. FNAL
  - work just started – aiming for spring/summer 2007
- requirements/goals for LCGO:
  - be at least as functional as existing systems (org.lcsim, GEAR, Mokka, SLIC,...)
  - enable smooth transition path from existing systems
  - encourage/increase interoperability between systems
  - have no known principle short comings: "everything should be possible"

ILC interoperable software chain



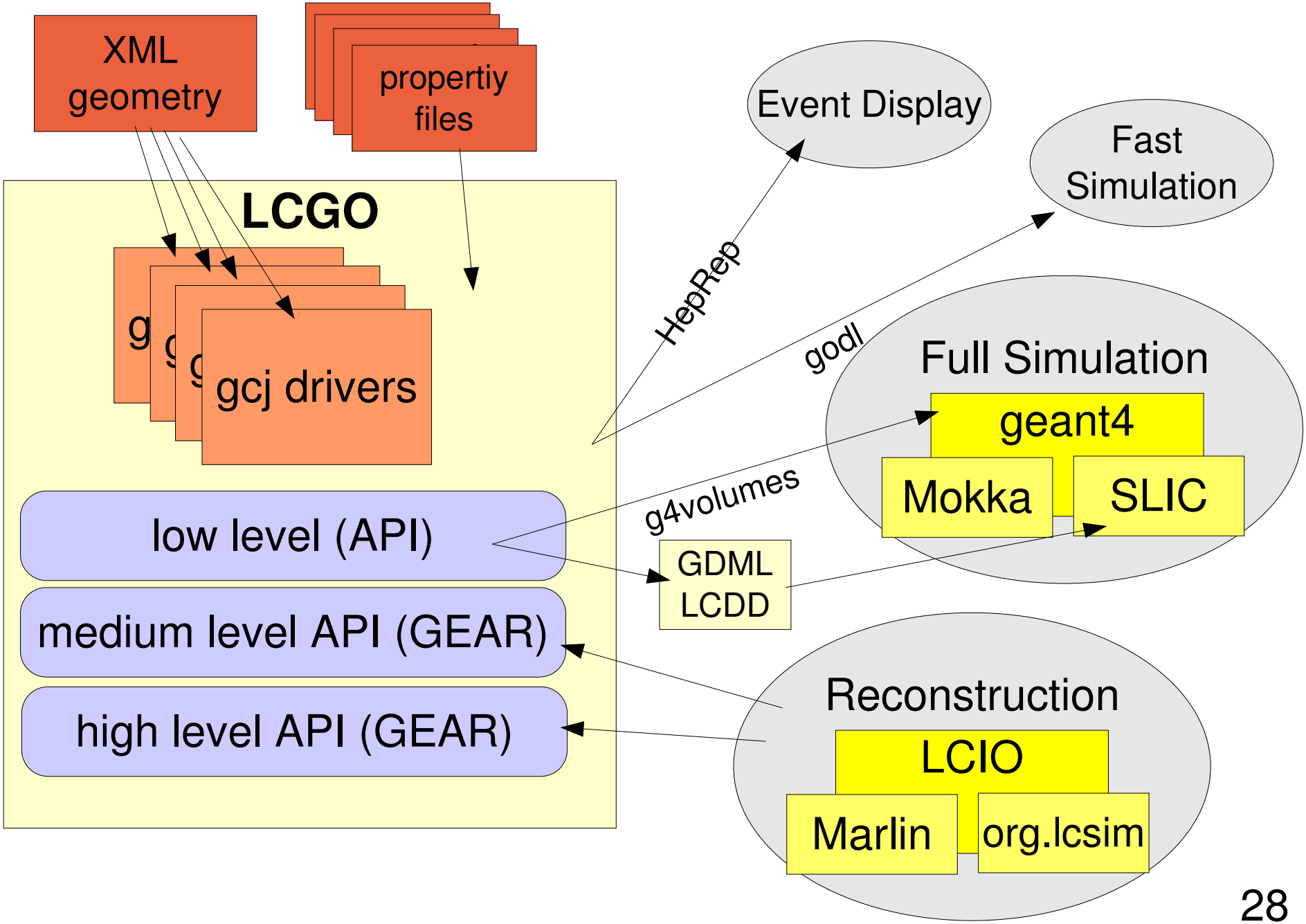
Summary

- EUDET task NA2 -ANALYS: "Provide a software framework for simulation and analysis (of testbeam data)"
  - developer hired at DESY (18 month)
- EUDET software development in context of existing ILC/LDC software framework
  - LCIO, Marlin, LCCD, GEAR, (LCGO) ...
- already used by some testbeam activities

All EUDET software activities
should be carried out in the context of
the existing software framework/ grid installations
e.g. DAQ software should use LCIO

Backup Slides

LCGO implementation prelim.



Grid activities

- ongoing grid activities (not related to EUDET):
  - DESY
    - H1and ZEUS Monte Carlo production
    - Tier2 for Atlas and CMS
    - ILC Monte Carlo production started
  - (ALU-Fr)
    - Tier2 for Atlas
- use existing experience to create grid infrastructure:
  - job submission scripts (computing grid)
  - data catalogue (data grid)

Plan: mainly RFWU-Bonn (postdoc) activity
in close collaboration with DESY groups (FLC,IT)

Deliverables and Requirements

- **requirements**:
  - documentation and its regular update are of utmost importance
    - “spread the information”
  - other EUDET participants should contribute by:
    - properly defining the *requirements* of the framework
    - *providing* and interfacing *simulation and reconstruction* software for the various detector technologies
    - testing the framework.
  - **deliverables**:
    - we expect to have a **first version** of the common data analysis and simulation framework ready **after 18 month**
    - **-> already available**
    - development however must continue throughout the whole duration of the project to cope with

(from annex1)

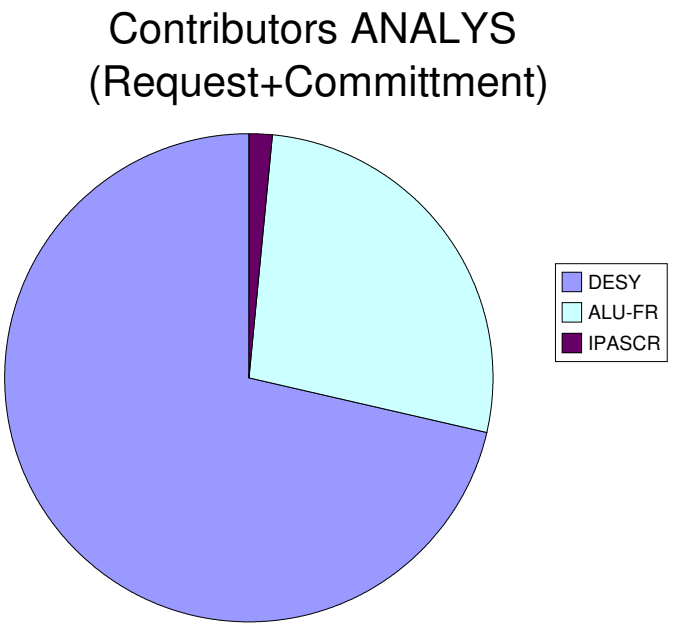
some LCGO planned features

- extended GEAR interface (medium and high level)
  - tracking (and clustering PFA)
    - average material volumes
    - intersection with 'next' volume
    - dE/dx
    - field maps
    - access to volumes
    - extensions of detectors (a la gear)
      - e.g. #layers, thickness, width,..
- material database
- field maps
- properties (sampling fractions)
- readout properties
  - cellId <-> position
  - cellid range (noise simulation)
  - cell sizes
  - neighbors
- Vector and Matrix classes ?
  - ThreeVector, Point3D
  - Planes, cylinders, ... ?
  - FourVector
  - SymMatrix (covariances)

Contributors for task ANALYS

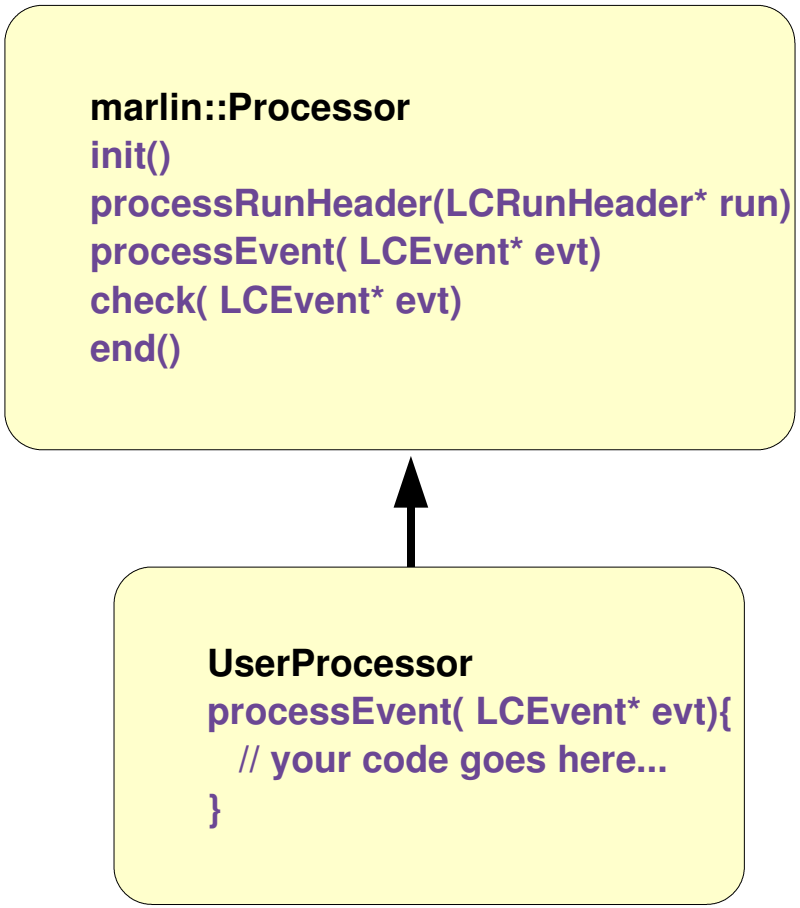
	DESY	ALU-FR	IPASCR	TOTAL
REQUEST				
Perm Staff ppm				
Temp Staff ppm	12.000	8.000		20.000
Perm Staff Cost kEUR				
Temp Staff Cost kEUR	62.500	46.875		109.375
Travels kEUR	1.300	0.867		2.167
Consumables kEUR				
Overheads kEUR	12.760	9.548		22.308
Total Manpower ppm	12.000	8.000		20.000
Total Cost kEUR	76.560	57.290		133.850
COMMITTMENT				
Perm Staff ppm	12.000		3.000	15.000
Temp Staff ppm				
Perm Staff Cost kEUR	62.500		9.000	71.500
Temp Staff Cost kEUR				
Travels kEUR				
Consumables kEUR				
Overheads kEUR	12.500		1.800	14.300
Total Manpower ppm	12.000		3.000	15.000
Total Cost kEUR	75.000		10.800	85.800
TOTAL BUDGET				
Perm Staff ppm	12.000		3.000	15.000
Temp Staff ppm	12.000	8.000		20.000
Perm Staff Cost kEUR	62.500		9.000	71.500
Temp Staff Cost kEUR	62.500	46.875		109.375
Travels kEUR	1.300	0.867		2.167
Consumables kEUR				
Overheads kEUR	25.260	9.548	1.800	36.608
Total Manpower ppm	**24.000**	**8.000**	**3.000**	**35.000**
Total Cost kEUR	**151.560**	**57.290**	**10.800**	**219.650**

ALU-FR now RFWU-Bonn



Marlin Processor

- provides main user callbacks
- has own set of input parameters
  - int, float, string (single and arrays)
  - parameter description
- naturally modularizes the application
- order of processors is defined via steering file:
  - easy to exchange one or several modules w/o recompiling
  - can run the same processor with different parameter set in one job
- processor task can be as simple as creating one histogram or as complex as track finding and fitting in the central tracker



Marlin core features

- fully configurable through steering files:
  - program flow
  - input parameters (processor based and global)
- self-documenting:
  - ./bin/Marlin -x
    prints example steering file with
    all available processors with their parameters and example/default values
- AIDA interface for histogramming
  - easy creation of histograms through absract interface
  - AIDAJNI/JAIDA, RAIDA (root based), ...
- configurable output
  - drop collections by name/type
- simple examples
  - user processor template, GNUmakefile,...
- easily extensible
  - makefiles 'automatically' include user packages with processors