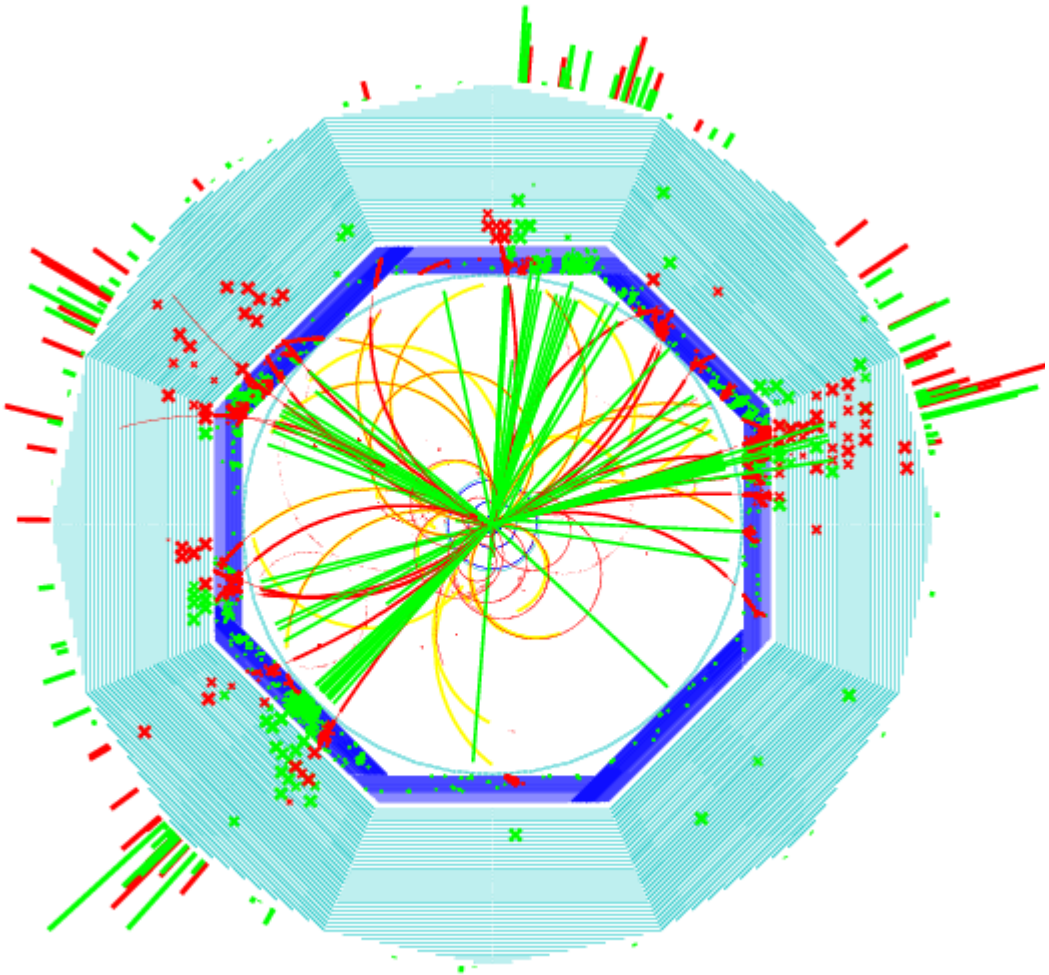
LC Software in the EU

Ties Behnke, DESY

- Simulation
- Reconstruction and related tools
- Geometries
- Summary and Outlook



Simulation

Main Full Simulation tools:

- MOKKA (GEANT4 based)
- BRAHMS (GEANT3 based)

The transition GEANT3 -> GEANT4 is in full swing

BRAHMS is still being used, and is still useful

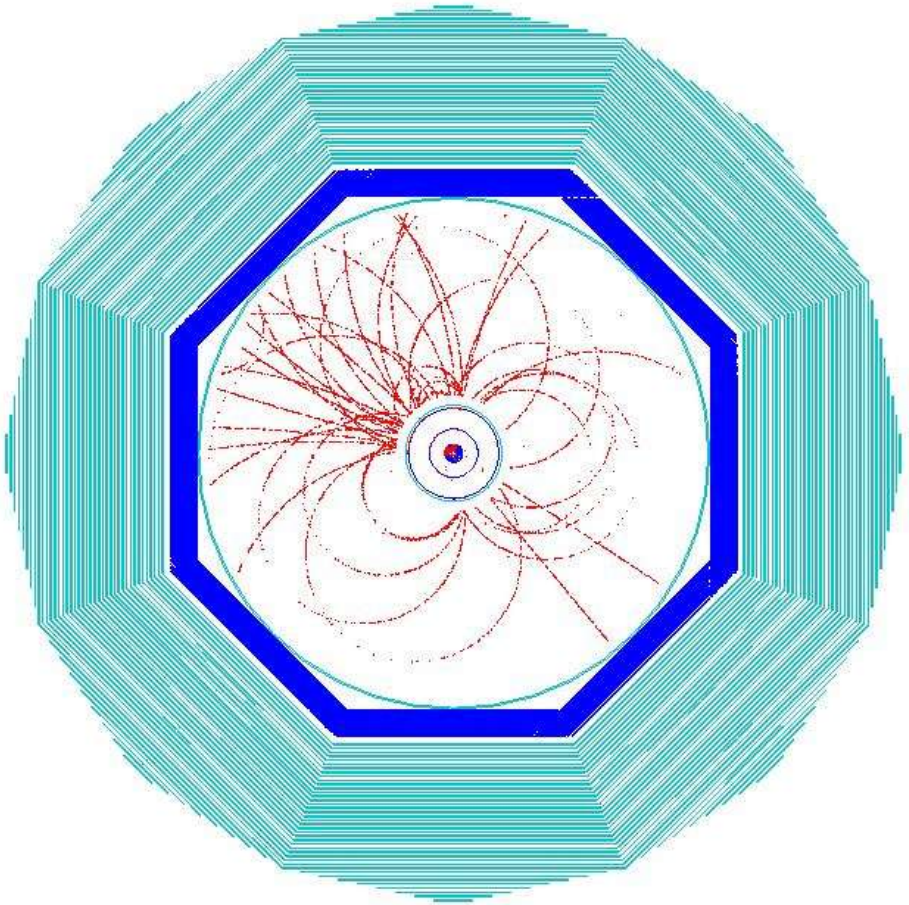
Most new developments are concentrated on MOKKA

Encouraging: The number of contributors to MOKKA is increasing!

Discussion: Continue (?) the move towards a more open software like development environment for MOKKA

Tracking Hits

BRAHMS simulation and reconstruction



Commands:

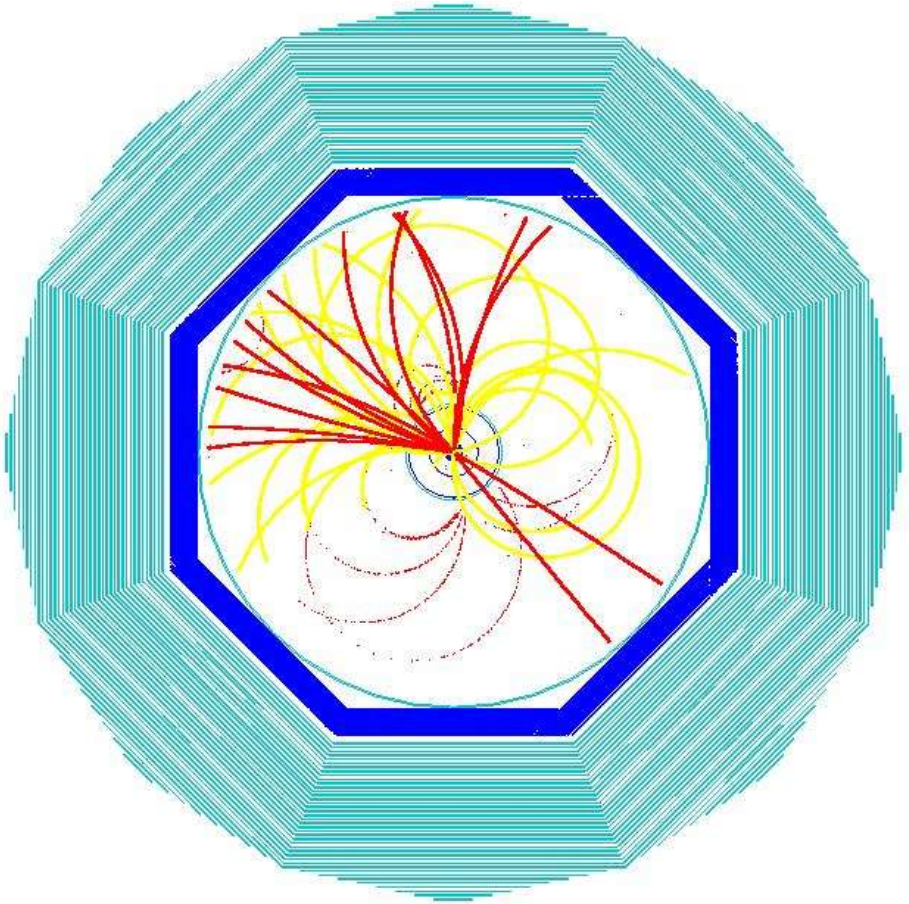
reco/cdet/hits 2 0.003

colour: 2 (red)
size: 0.003

Fully Reconstructed Event

Commands:

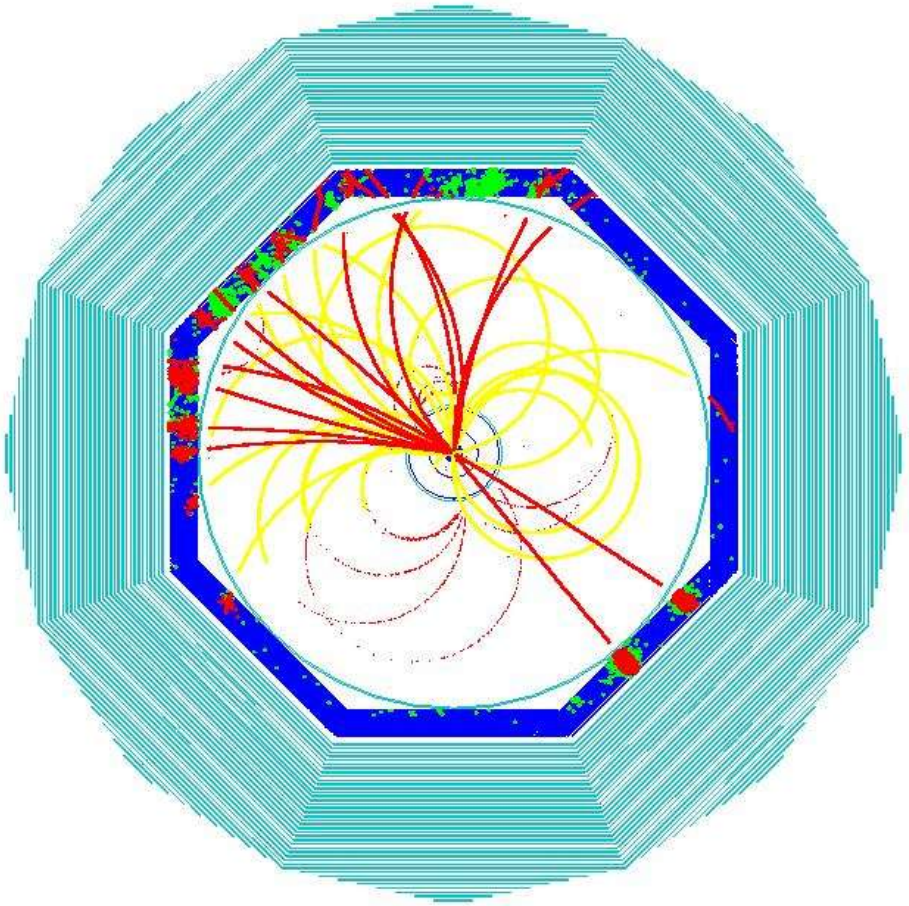
reco/cdet/hits 2 0.003
reco/cdet/track 0 2 2.0 0 3



Fully Reconstructed Event

Commands:

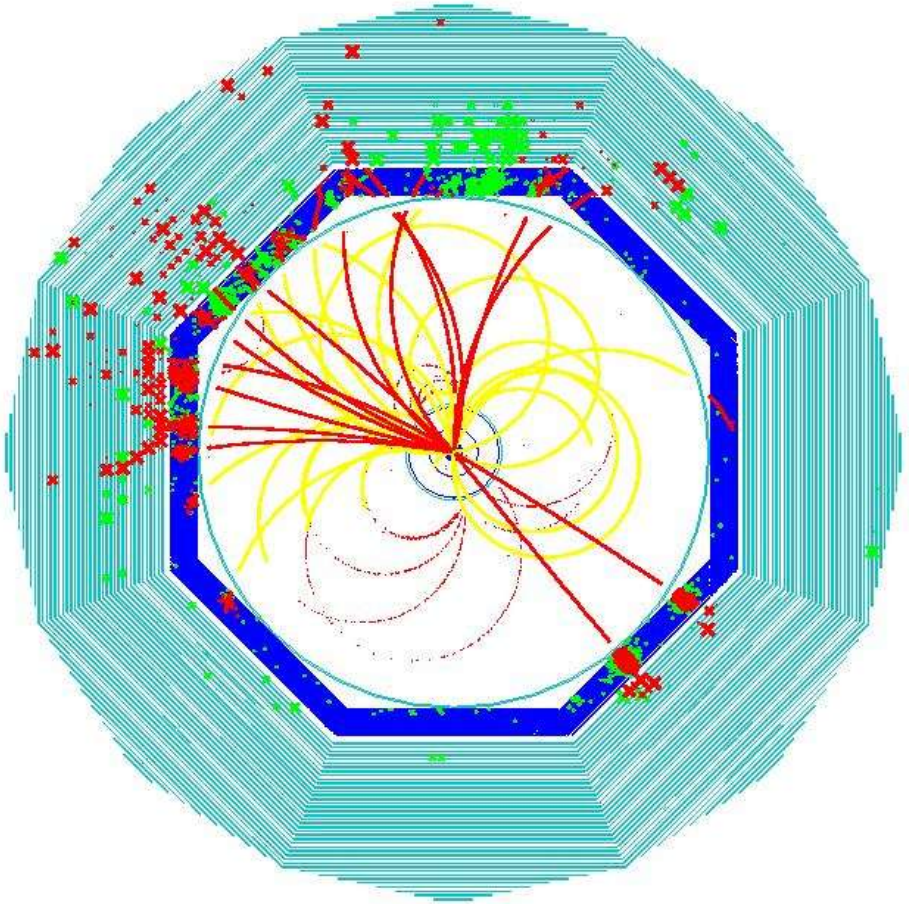
```
reco/cdet/hits 2 0.003
reco/cdet/track 0 2 2.0 0 3
reco/calo/ecal/assigned 2 0.03 260.
reco/calo/ecal/cluster 3 0.03 260.
```



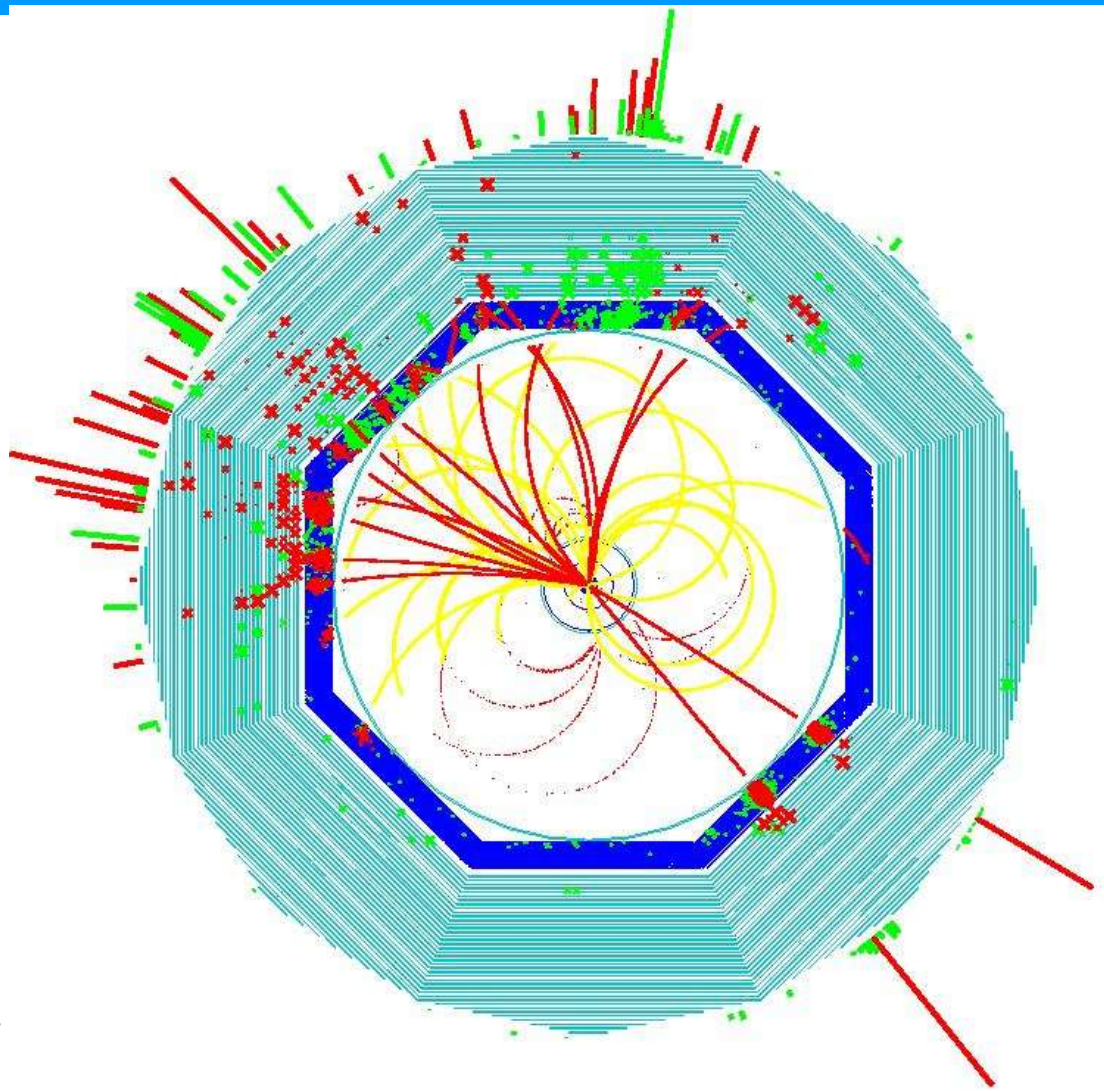
Fully Reconstructed Event

Commands:

```
reco/cdet/hits 2 0.003
reco/cdet/track 0 2 2.0 0 3
reco/calo/ecal/assigned 2 0.03 260.
reco/calo/ecal/cluster 3 0.03 260.
reco/calo/hcal/size 2 0.0 100 260.
reco/calo/hcal/cluster 3 0.0 100 260.
```



Fully Reconstructed Event

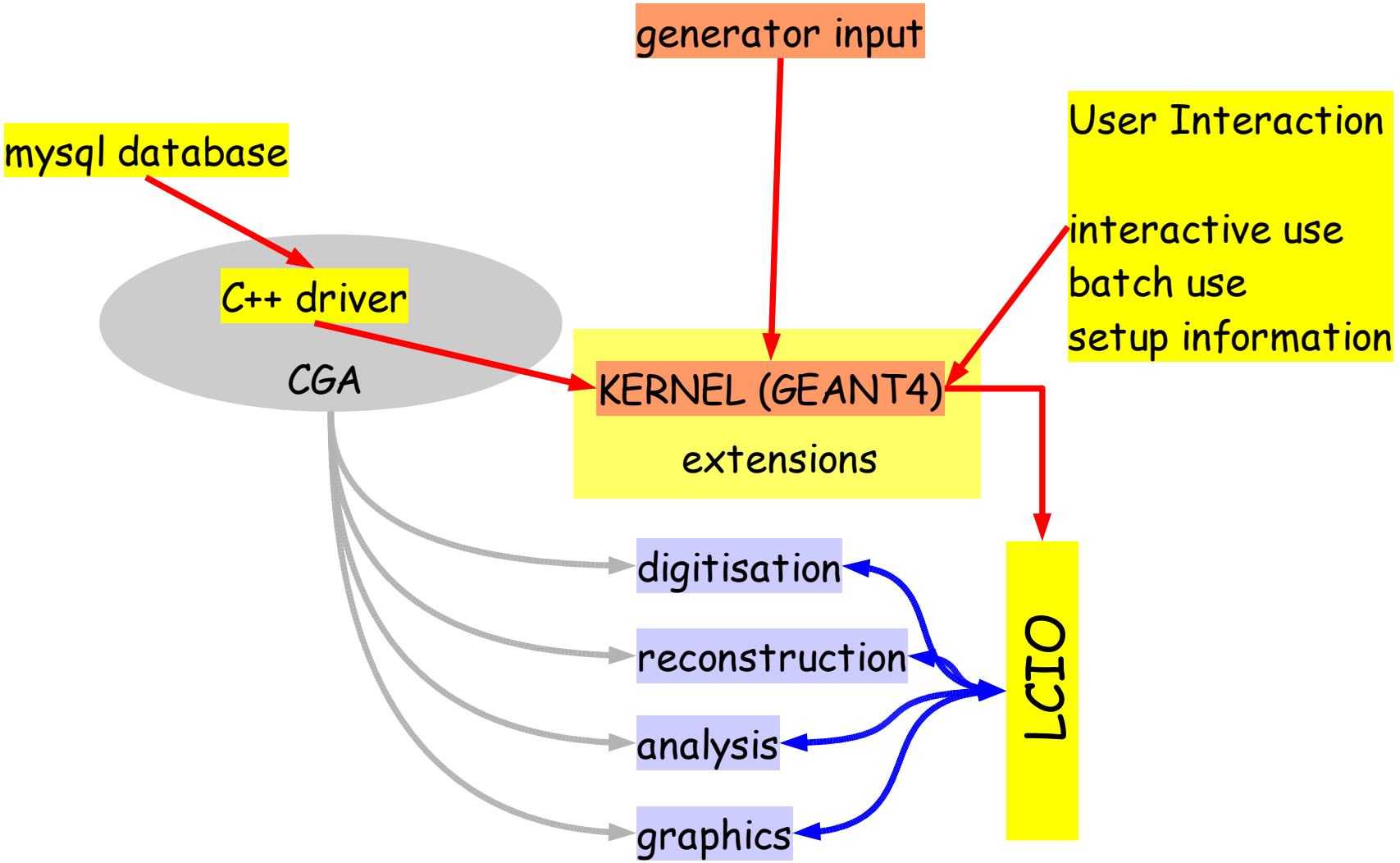


Commands:

```
reco/cdet/hits 2 0.003
reco/cdet/track 0 2 2.0 0 3
reco/calo/ecal/assigned 2 0.03 260.
reco/calo/ecal/cluster 3 0.03 260.
reco/calo/hcal/size 2 0.0 100 260.
reco/calo/hcal/cluster 3 0.0 100 260.
reco/eflow/out 3 2 5
```

BRAHMS is available from the ECFA simulation homepage for download

MOKKA Design



CGA: common geometry access

MOKKA: Status

available:

- full fledged and detailed implementation of different TESLA like detectors
- detailed model of the CALICE test calorimeter
- interface to STDHEP (binary and ascii)
- control via command line and steering file
- output to LCIO (latest version)

MOKKA can be used for detailed simulation studies of the LC detector

New geometries are simple to implement (but need to write some C++ code)

MOKKA: Open Questions

Continue to improve the sub-detectors

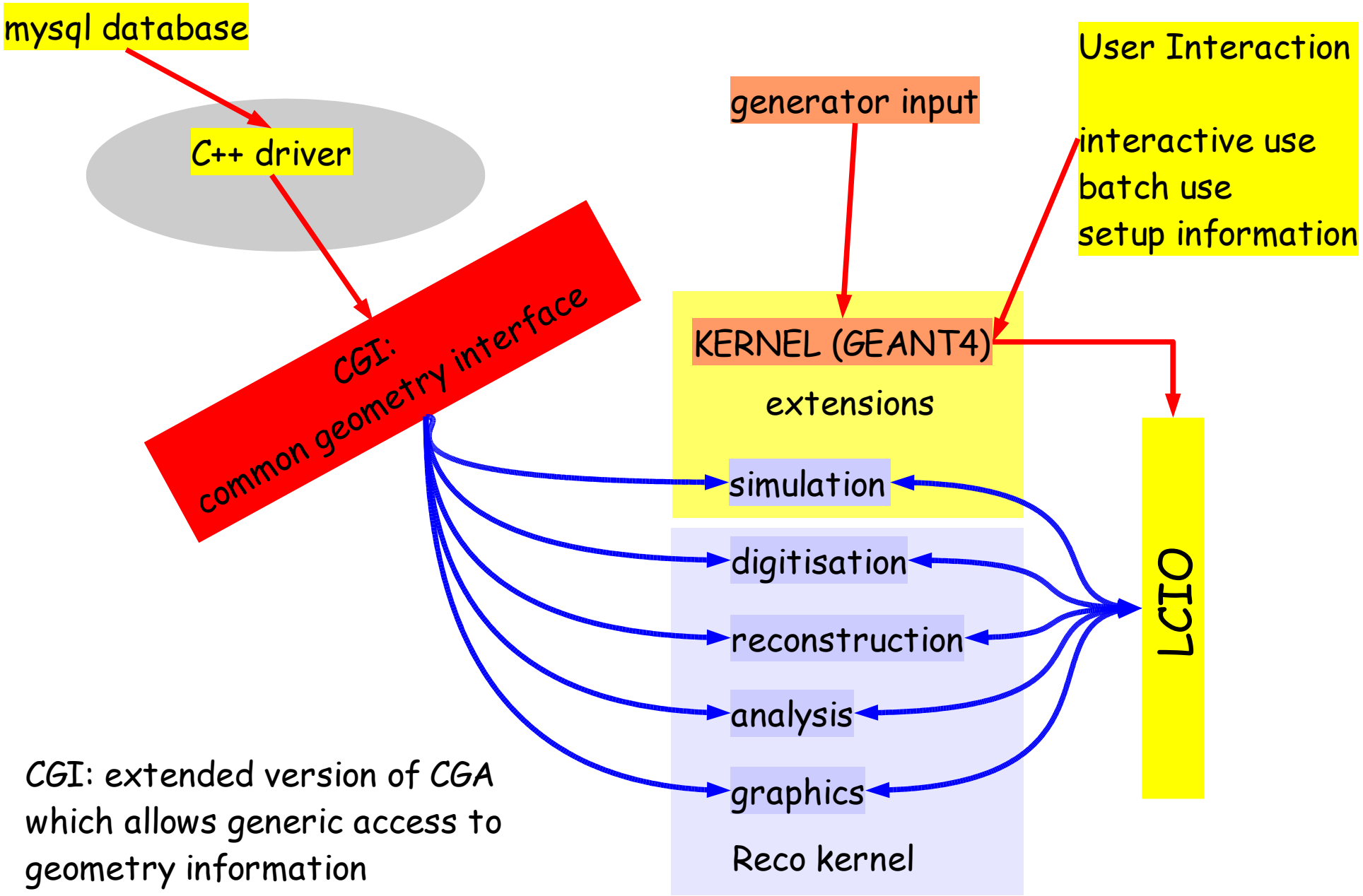
work is needed in particular in the area of tracking detectors

Continue to discuss and develop the way the database is used, in particular during development (shared variables? satelite databases? private databases? support for multiple databases? ....)

Conceptually we have adopted the database approach for the short and medium term developments.

note: we still maintain an interest in providing an interface to the XML based geometry systems used and developed in the US

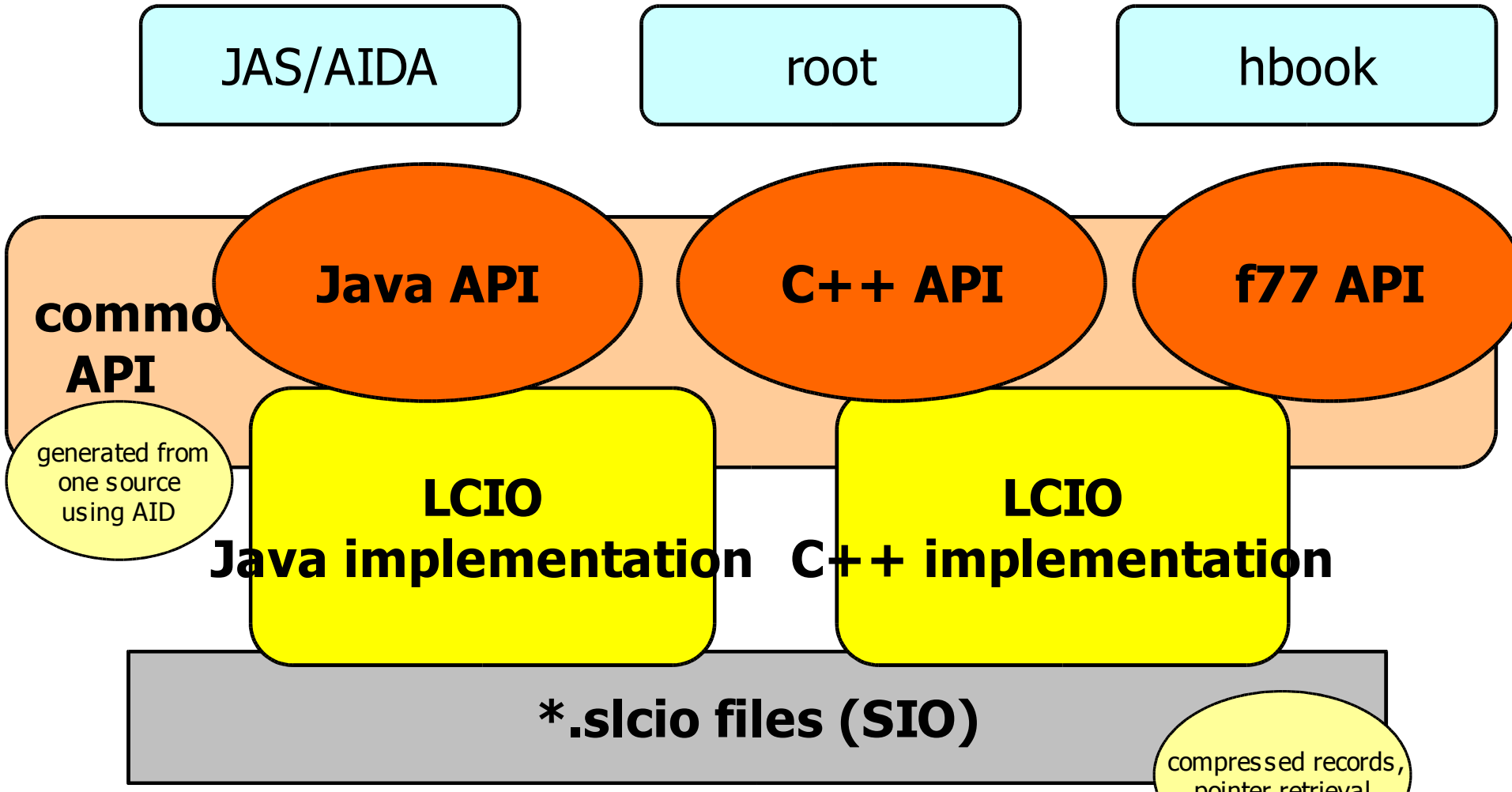
MOKKA Design: future?



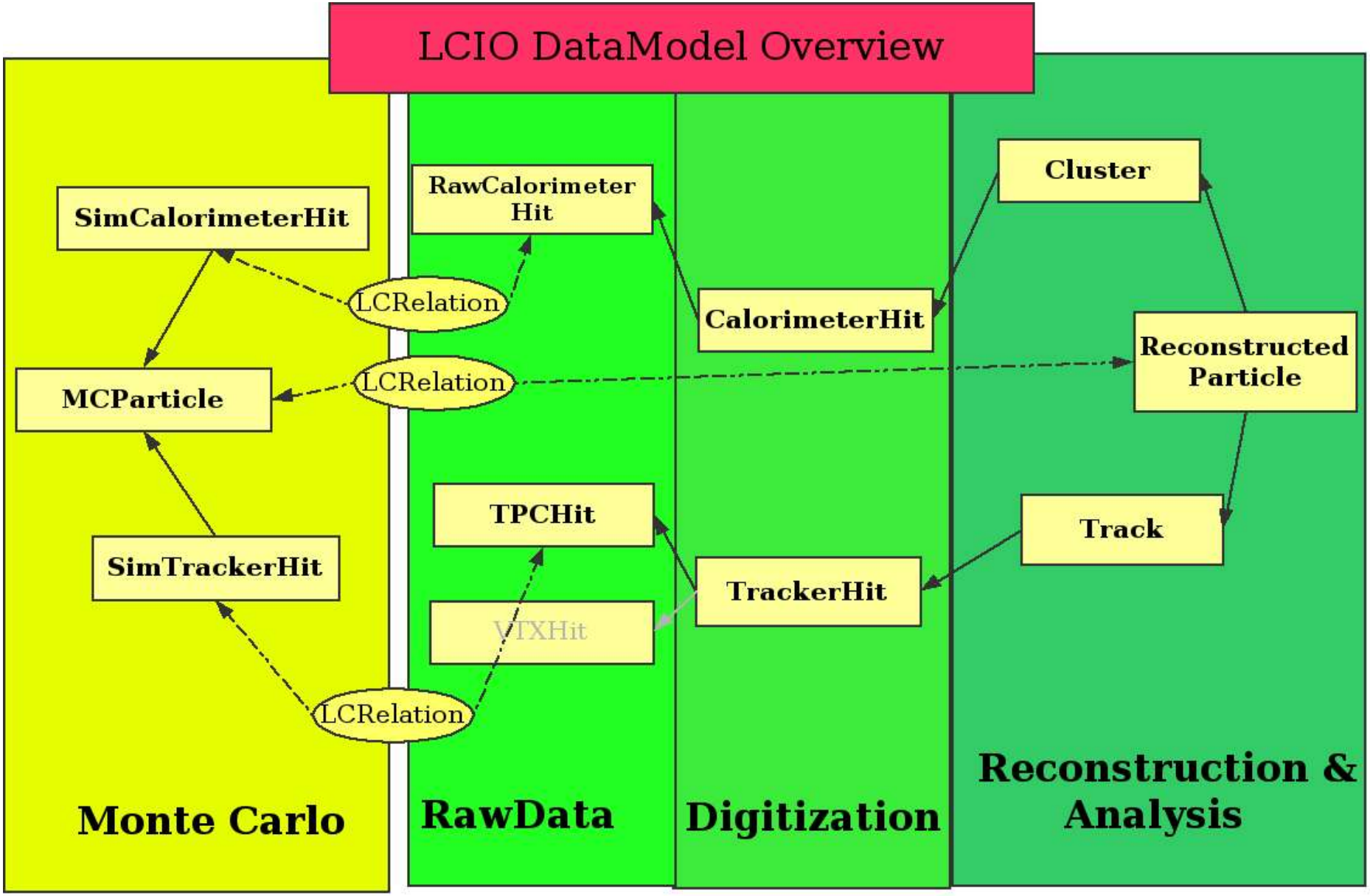
CGI: extended version of CGA which allows generic access to geometry information

LCIO SW-Architecture

LCIO: common LC input output format and data model for LC studies:
DESY- LLR Paris – SLAC (main contributors Tony Johnson, Frank Gaede)



Data Model


LCIO DataModel Overview
SimCalorimeterHit
RawCalorimeter Hit
Cluster
LCRelation
CalorimeterHit
Reconstructed Particle
LCRelation
MCParticle
TPCHit
Track
SimTrackerHit
TrackerHit
VTXHit
LCRelation
Monte Carlo
RawData
Digitization
Reconstruction & Analysis

The data model

important ingredients:

- objects (tracks, clusters, ...) are grouped into collections
- there can be several collections of the same type of objects in the event:
  tracks at IP
  tracks at Calo face
  VTX tracks
  ...

  (if this is done, documentation is essential!)
- self-referencing of the objects allows the buildup of tree structures

LCIO Customers/Users

- Mokka simulation (see talk)
- Brahms reconstruction (see talk)
- JAS3
  - provides convenient file browser
  - will have LCIO-WIRED plugin -> generic event display !
- Calorimeter group (DESY)
  - has MiniCal raw data converted to LCIO files
  - to be used also for Hcal physics prototype
- TPC groups (DESY & Aachen & ...)
  - will use LCIO for prototype
- Lelaps fast Monte Carlo
- hep.lcd reconstruction: ongoing
- other groups looking into using LCIO

General Software Framework

Proposed basic guidelines:

All software from simulation to reconstruction:

- is based on the LCIO data model
- uses LCIO as a persistency mechanism
- uses LCIO as a transient data model between modules

A de-facto standard in Europe: core software (simulation, reconstruction) is based on plain C++

- no root dependencies in any central software
- other languages, if used, are integrated through wrappers from C++

We are very interested to provide
a close integration of JAVA into this,
but the technical side is not really solved.

The user backend is totally open: Root, JAS, PAW, ...

Reconstruction

- First "reconstruction framework" exists: MARLIN

  Modular Analysis and Reconstruction for the LiNear Collider

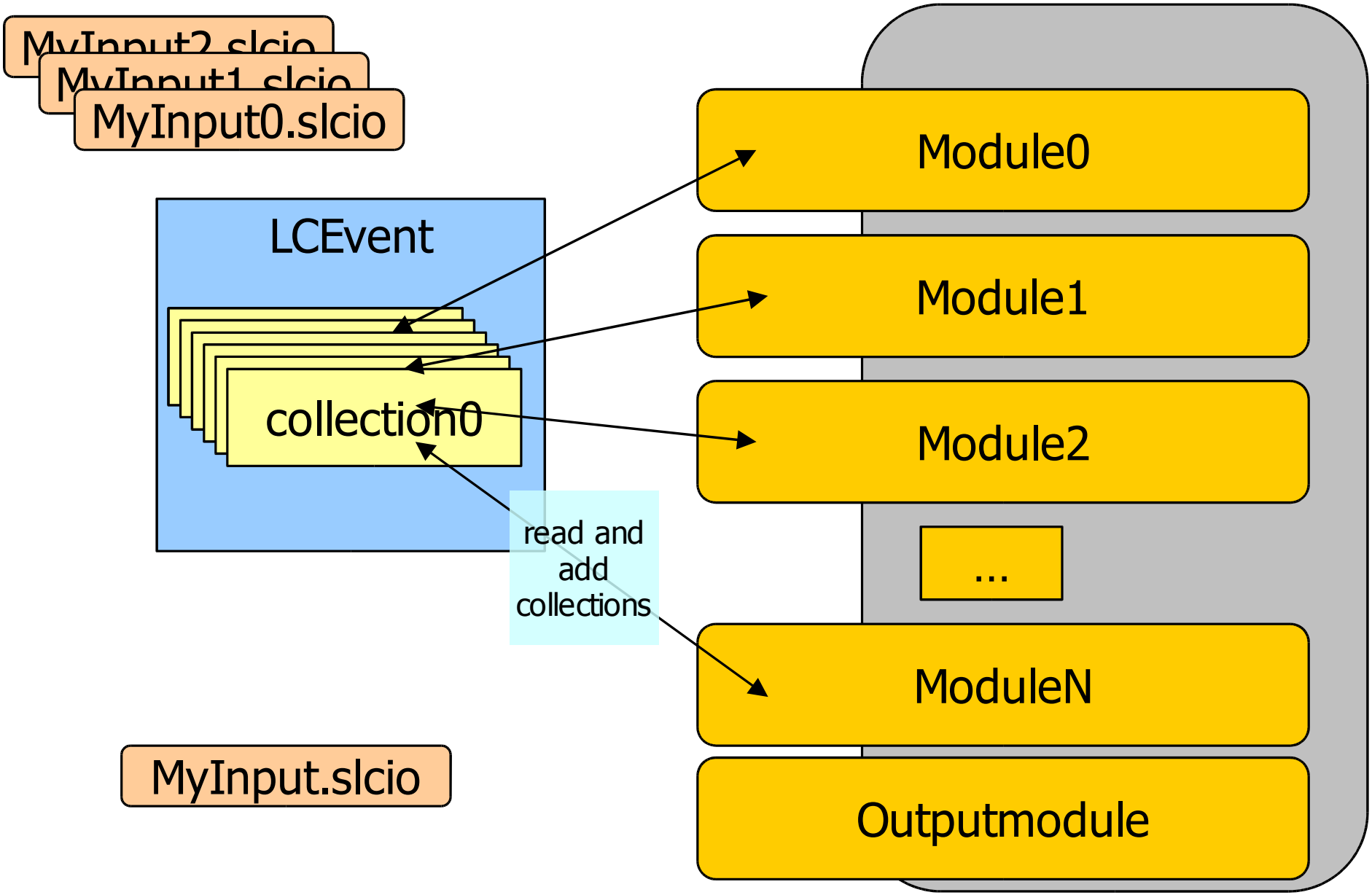
- see talk by J. Samson in this session

- simple, open framework
  - dynamically configured through steering file
  - defines a standard structure for a module
  - LCIO based

- Its a starting point, lots still needs to be done

existing modules:
HCAL prototype ganging module
Jet Finder, Lepton Finder, ZVTOP module
soon: wrapped reconstruction software (tracking, ...)

Modules and the LCIOEvent



MarlinModule

- MarlinModule: base class for all user modules
- provides hooks (callbacks) for user actions:
  - init()
    - called once at program start
    - use to initialize histograms, counters, etc.
  - processRunHeader(LCRunHeader* run)
    - called for bookkeeping – new run conditions ?
  - processEvent(LCEvent* evt)
    - the working horse – this where the analysis takes place
  - end()
    - called once at end of job
    - write out histos, …

MARLIN developments

Under discussion:

try to make the user hooks as similar as possible to the ones in the JAVA (LCD) framework to facilitate exchange of ideas

A problem:

The true parallel use of JAVA and C++ code to access the same LCIO even in memory is difficult

We are still far from a truly language independent frame

Interfaces

- Need to develop common tools – the wider the user community, the better
- First attempts to agree on common programs have notalways converged
- Proposal: concentrate on the definition of interfaces

LCIO: interface between data and programs:
an example for a very successful collaboration (SLAC-DESY-LLR)

CGI: Common Geometry Interface

define access methods for the basic geometry items
examples:

- getmaterial
- getX0
- getposition
- ....

prepare a concrete proposal based on the existing CGA

Have to get the discussion started!

Data Access

Agreement on LCIO facilitates the sharing of data:

now we need to develop the tools to actually share the data

Possible approaches:

"stand-alone" client server architecture

example: the "old" US system integrated into JAS2

GRID based infrastructure

use GRID tools to access data transparently from different servers around the world: some first tests done at DESY

Work has not really started on this .... need to make a real effort in the near future to get things going.

Summary

We are making progress – some at least

- LCIO development is converging
- Several groups are using LCIO both in test experiments and in simulation
- MARLIN is starting point for a modern C++ based reconstruction system
- First tools for modern reconstruction framework in Europe start to appear