LDC Software

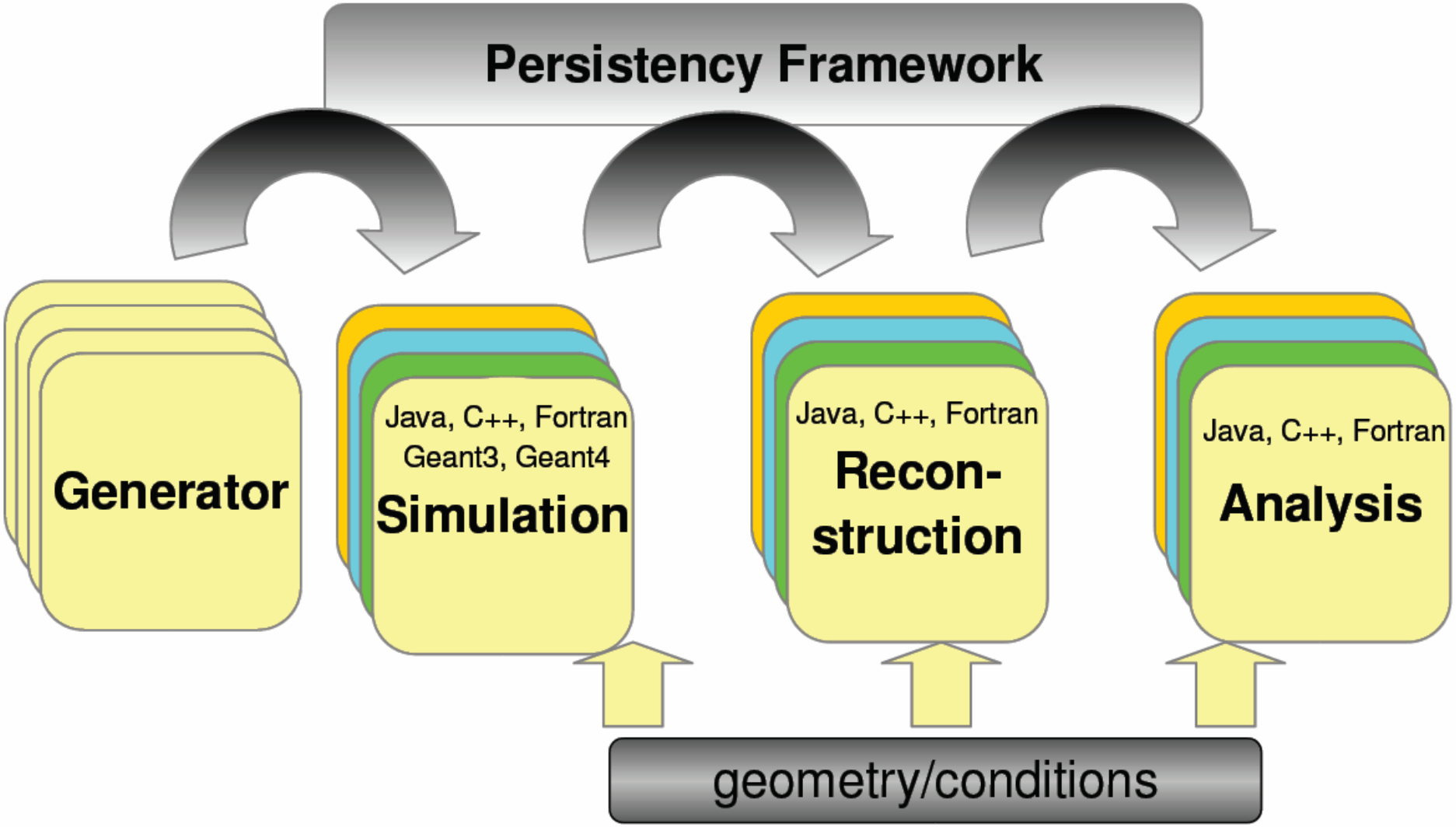
Georgios Mavromanolakis

University of Cambridge and Fermilab

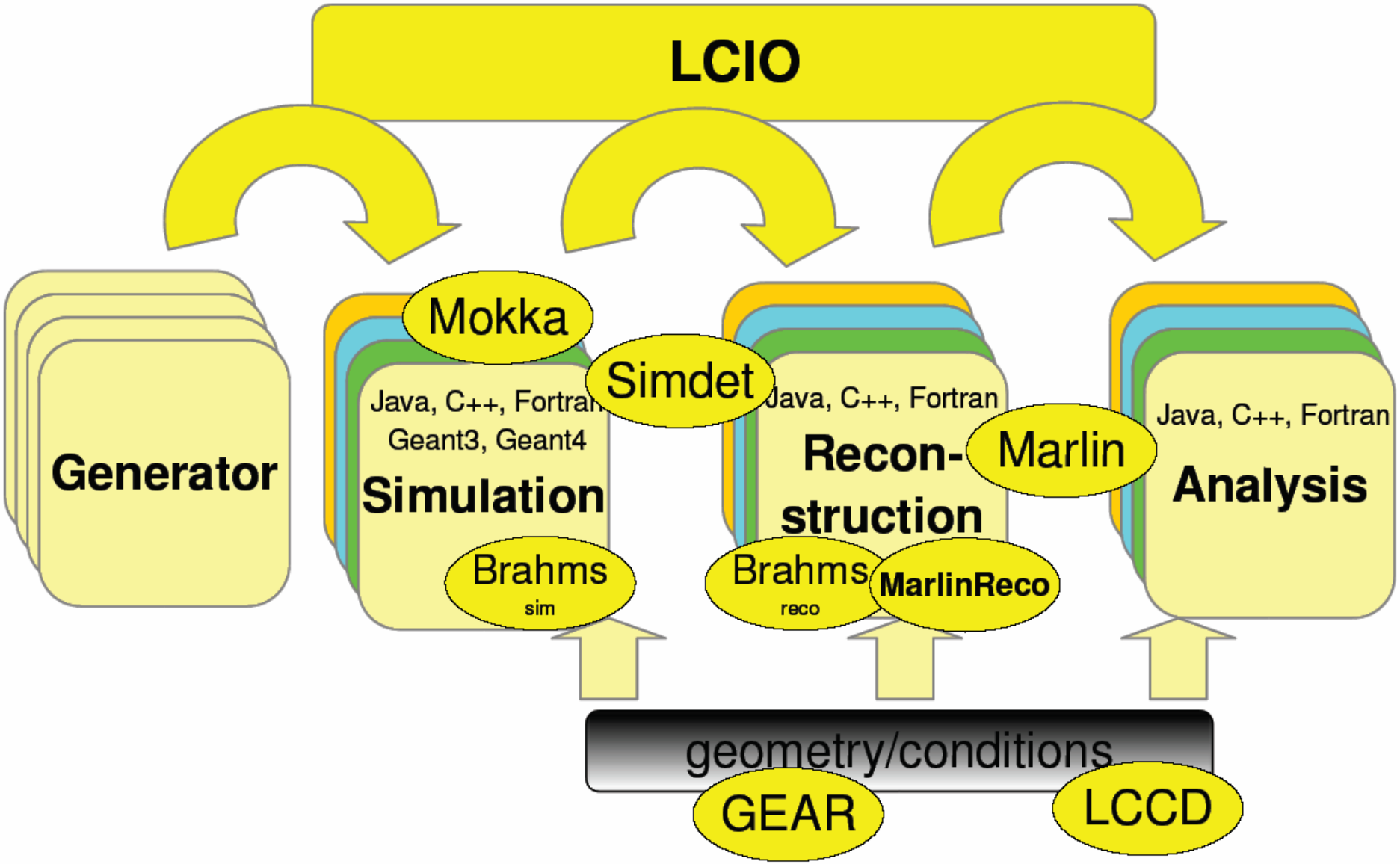
Outline

- General overview
- LCIO data model
- Marlin and processors
- Gear, LCCD, MarlinReco, etc
- Software dependencies
- Examples (a processor, a steering file, etc)

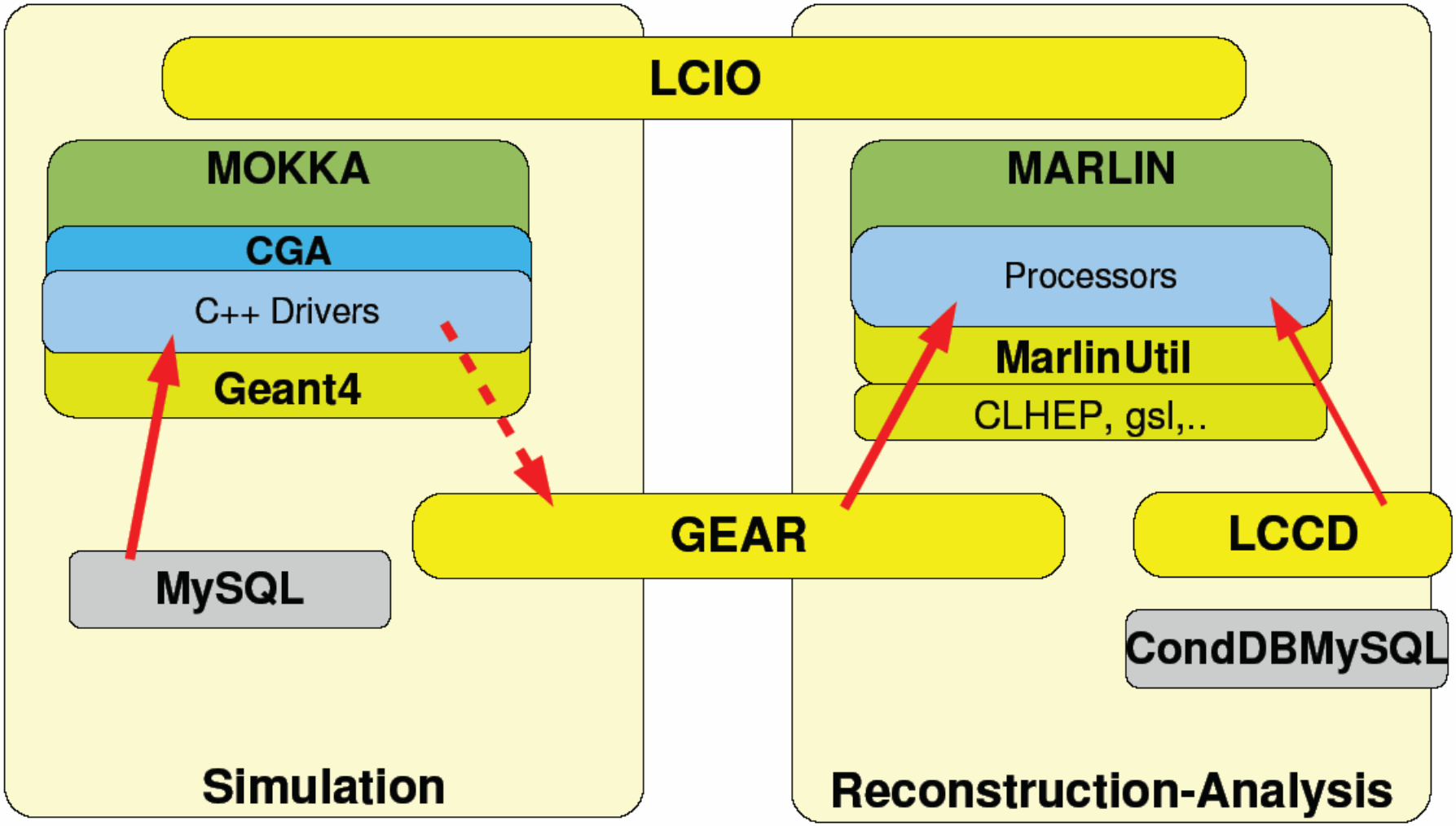
ILC software chain



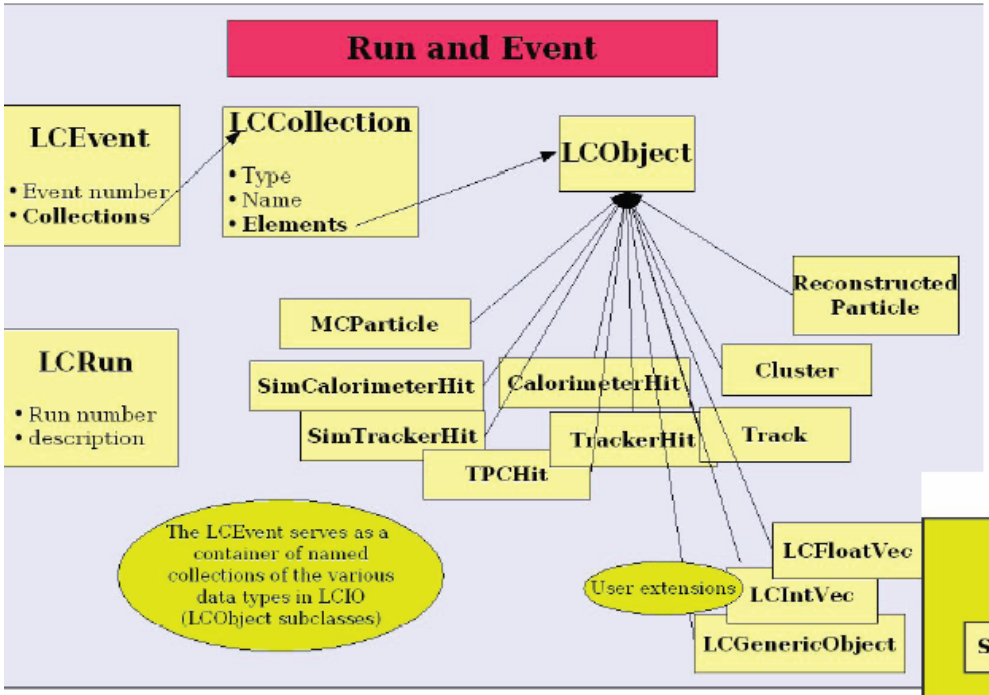
ILC software tools used for LDC



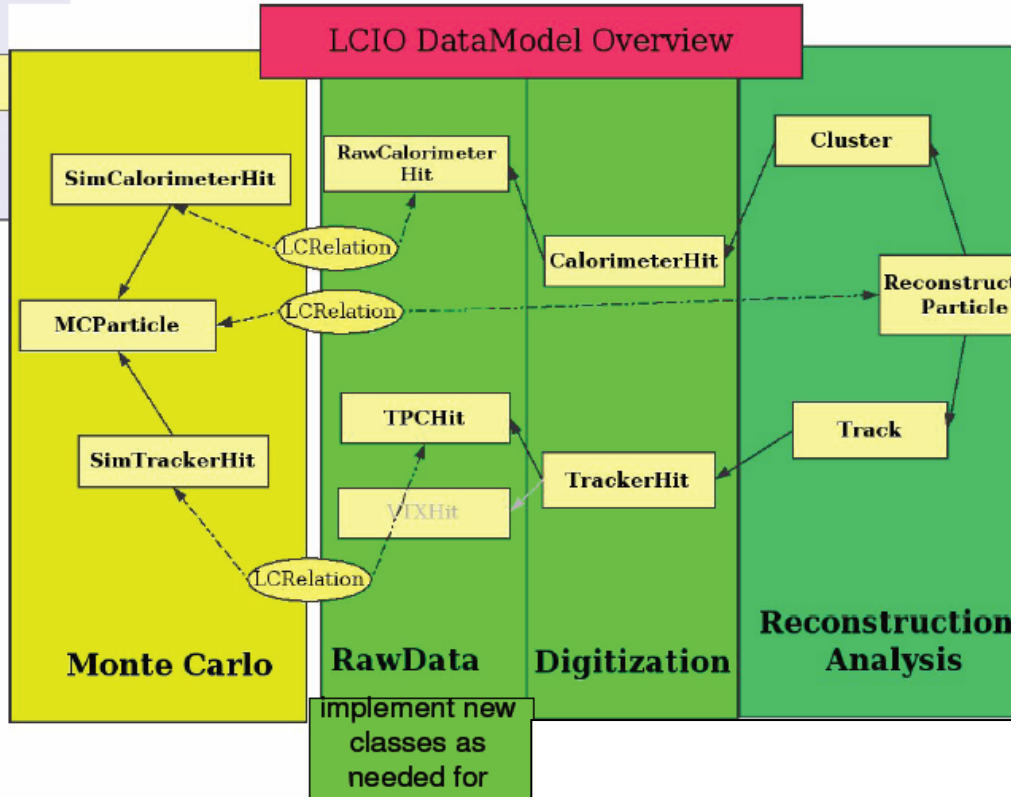
LDC simulation framework



LCIO data model

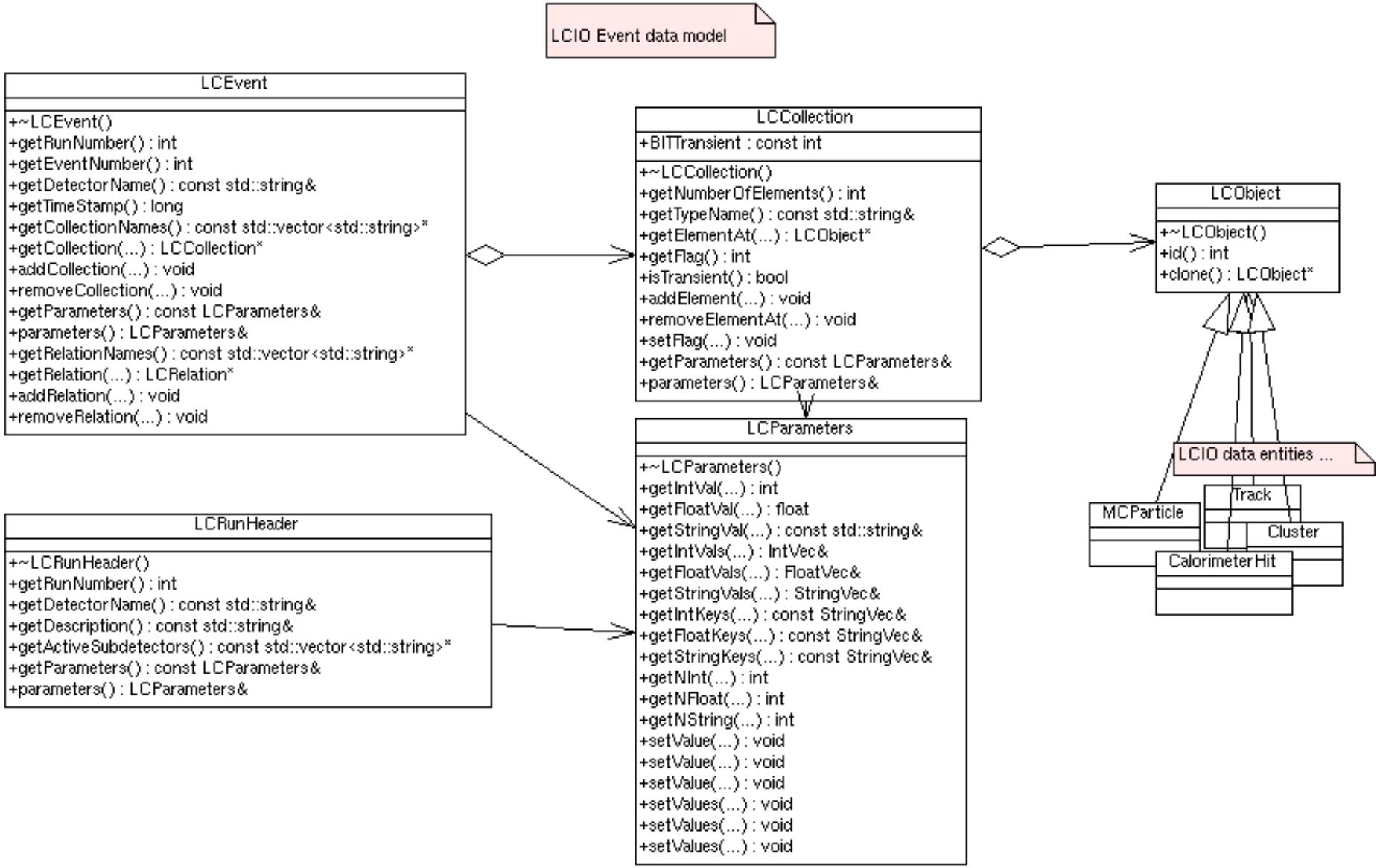


event serves as container of untyped collections

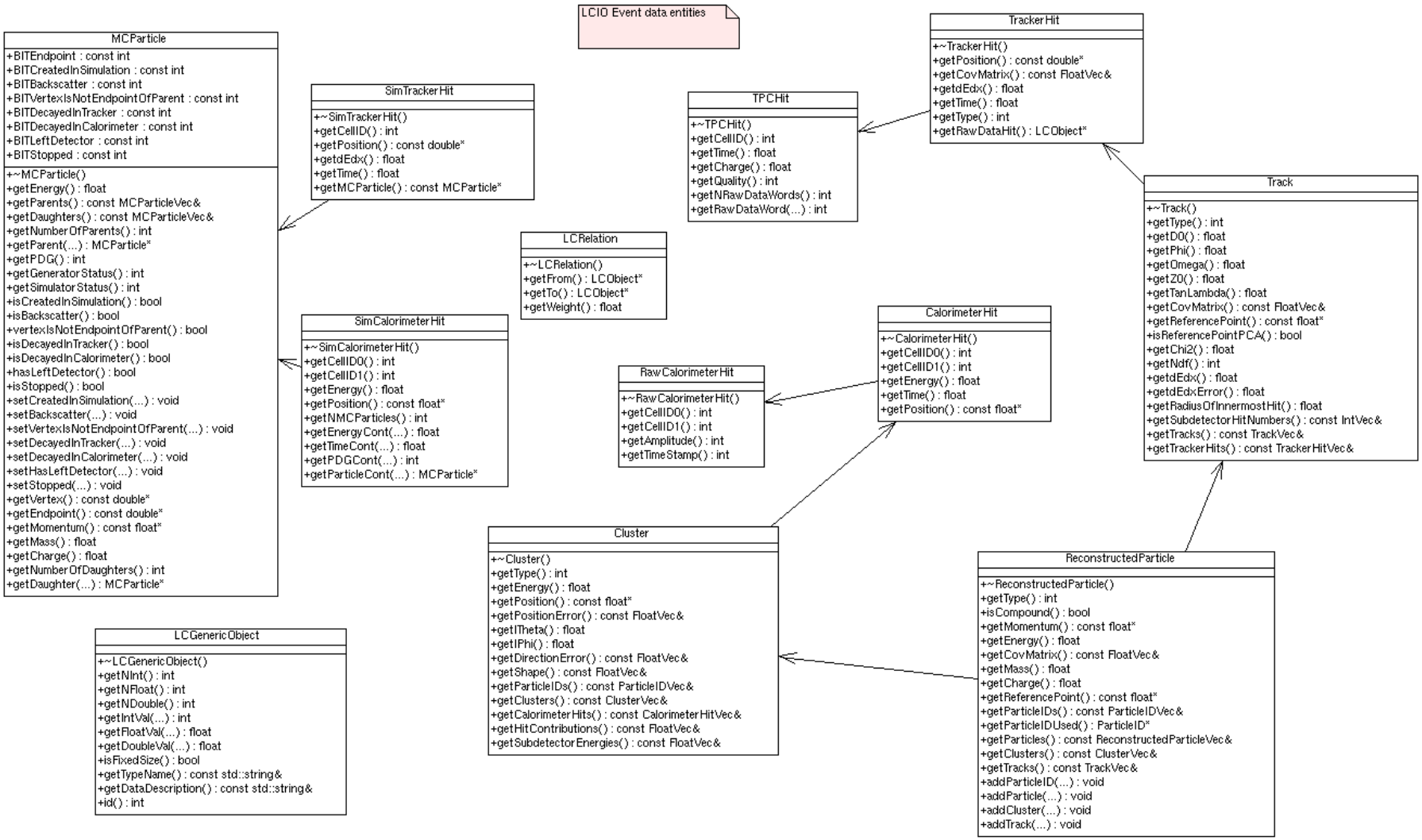


hierarchy of data objects in the event

LCIO Event data model



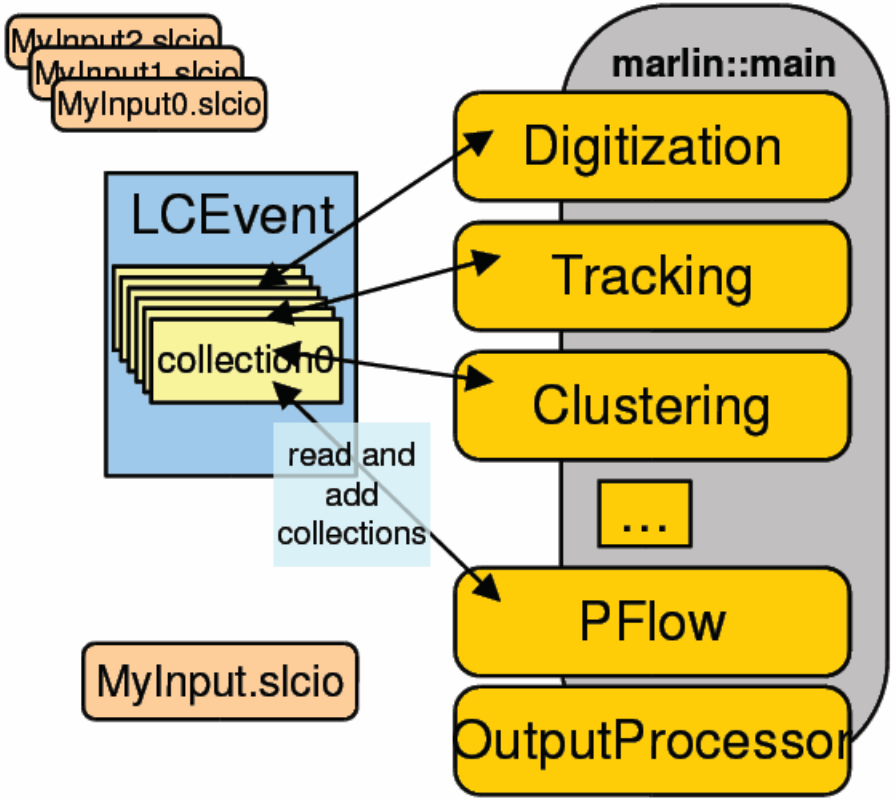
LCIO Event data entities



Marlin

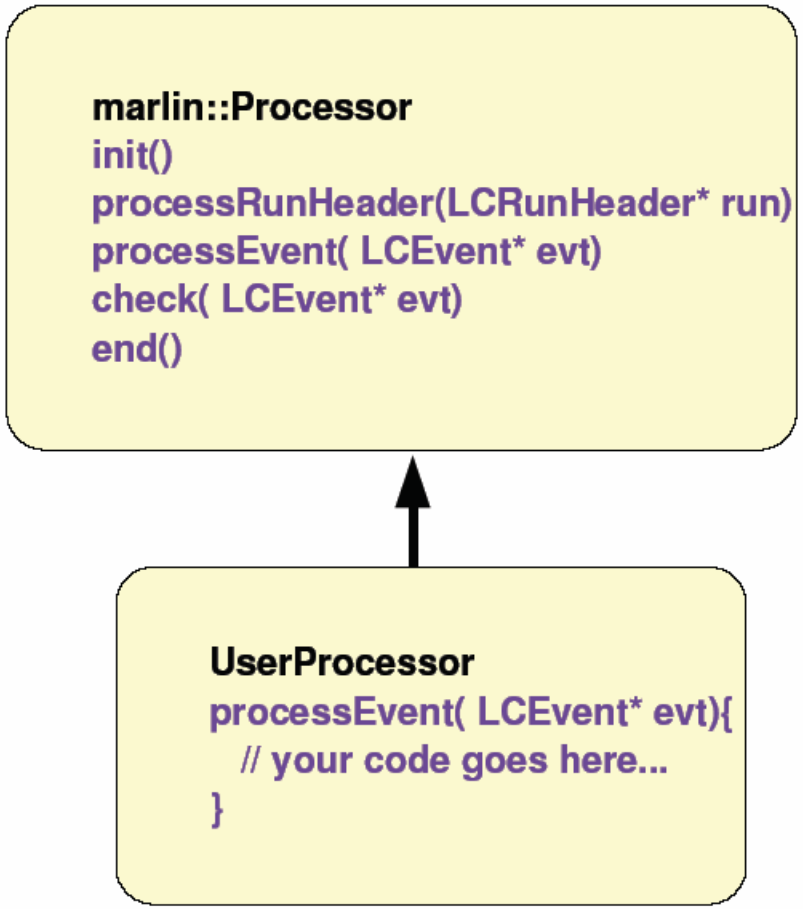
Modular**A**nalysis & **R**econstruction for the **L I N**ear Collider

- 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 Processor

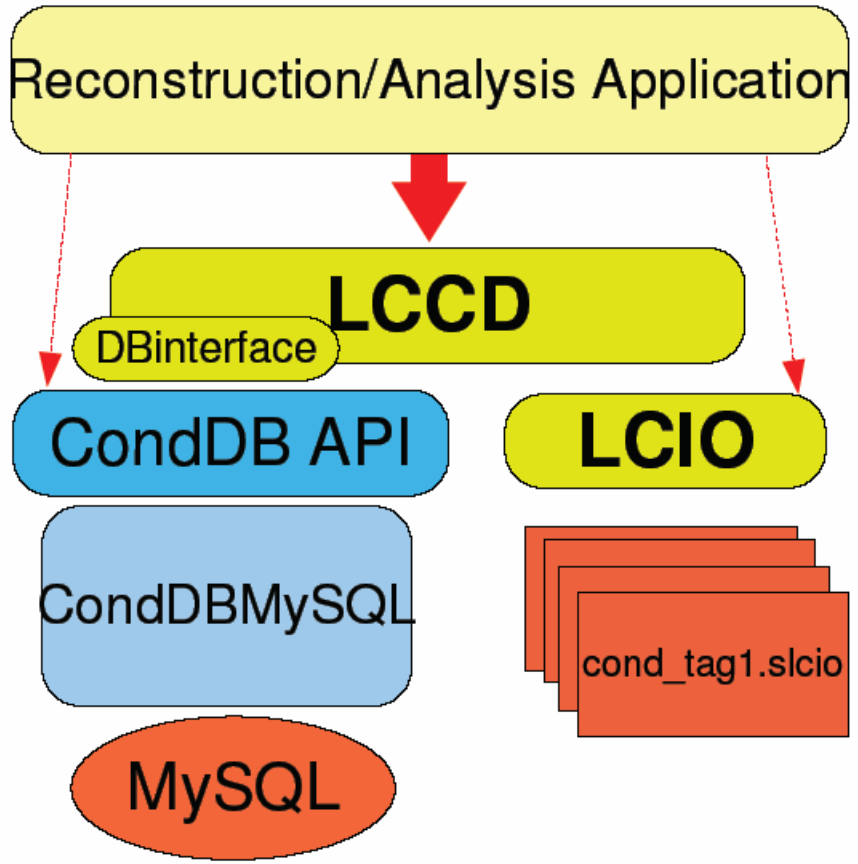
- provides main user callbacks
- has own set of input parameters
  - int, float, string (single and arrays)
  - parameter description
- naturally modularizes the application
- order of processors is defined via steering file:
  - easy to exchange one or several modules w/o recompiling
  - can run the 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



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" typ
    <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 fo reconstruction
  - provides access to material properties - planned
- abstract interface (a la LCIO)
- concrete implementation based on XML files
  - and Mokka-CGA – planned

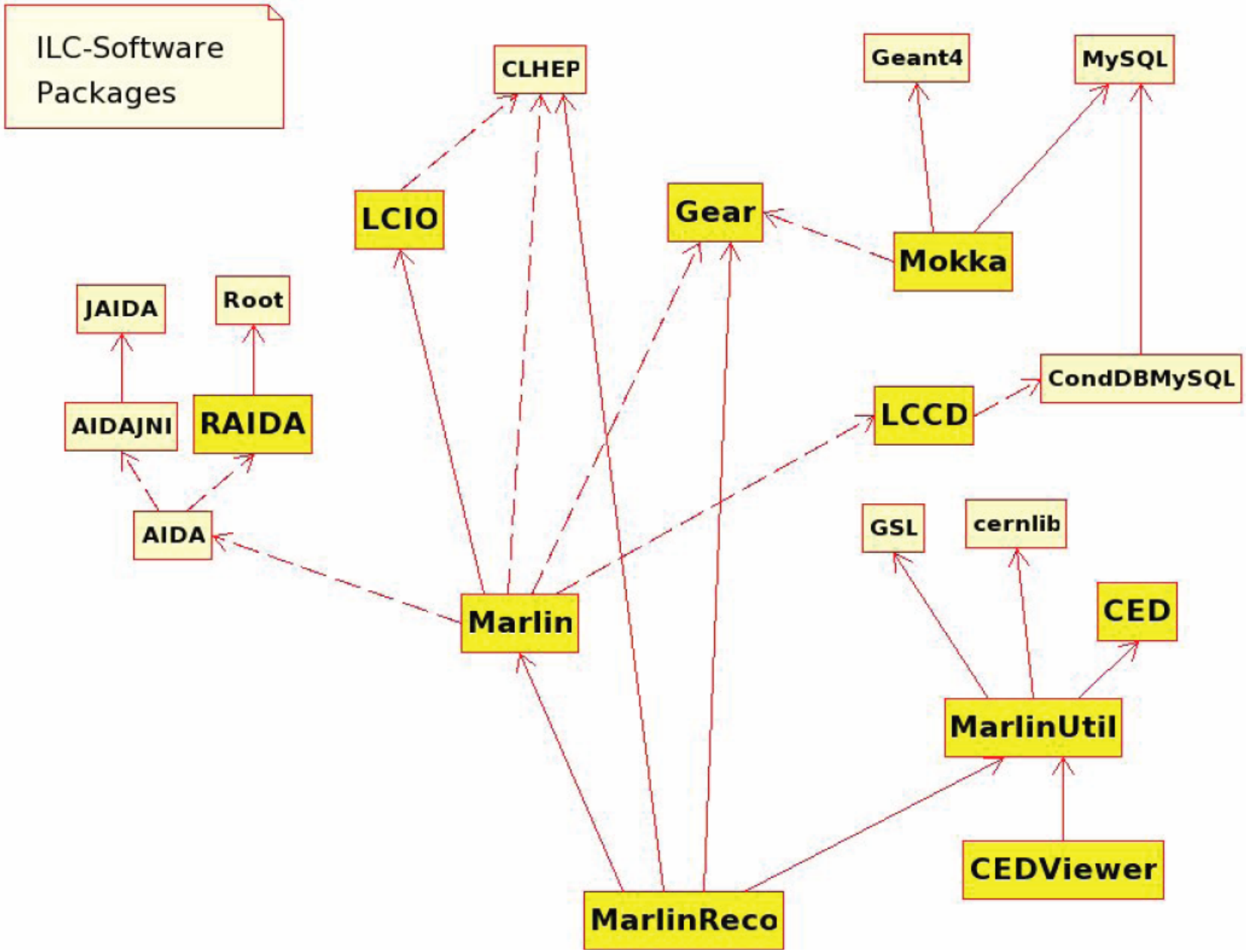
MarlinReco

A toolkit of processors to do digitization, tracking, clustering, particle flow analysis etc

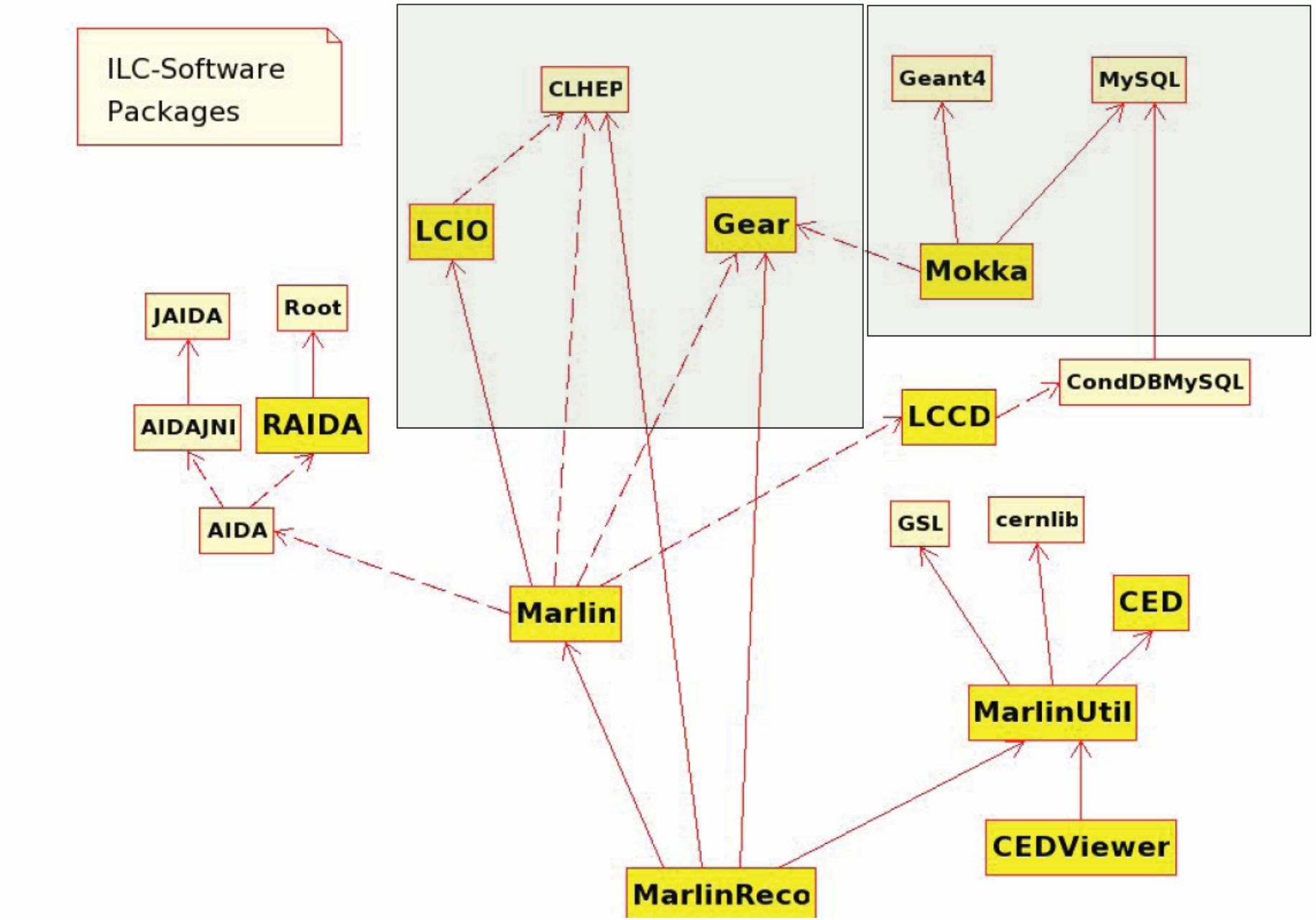
- Digitization
  - VTXDigi
  - TPCDigi
  - LDCCaloDigi
- Tracking
  - VTXTracking
  - LEPTracking
  - TrackCheater
- Clustering
  - TrackwiseClustering
  - ClusterCheater
- Pflow and analysis
  - Wolf
  - PandoraPFA
  - EventShapes
  - SatoruJetFinder
  - …

also MarlinUtil package with various utility processors to do event display, to produce persistent output

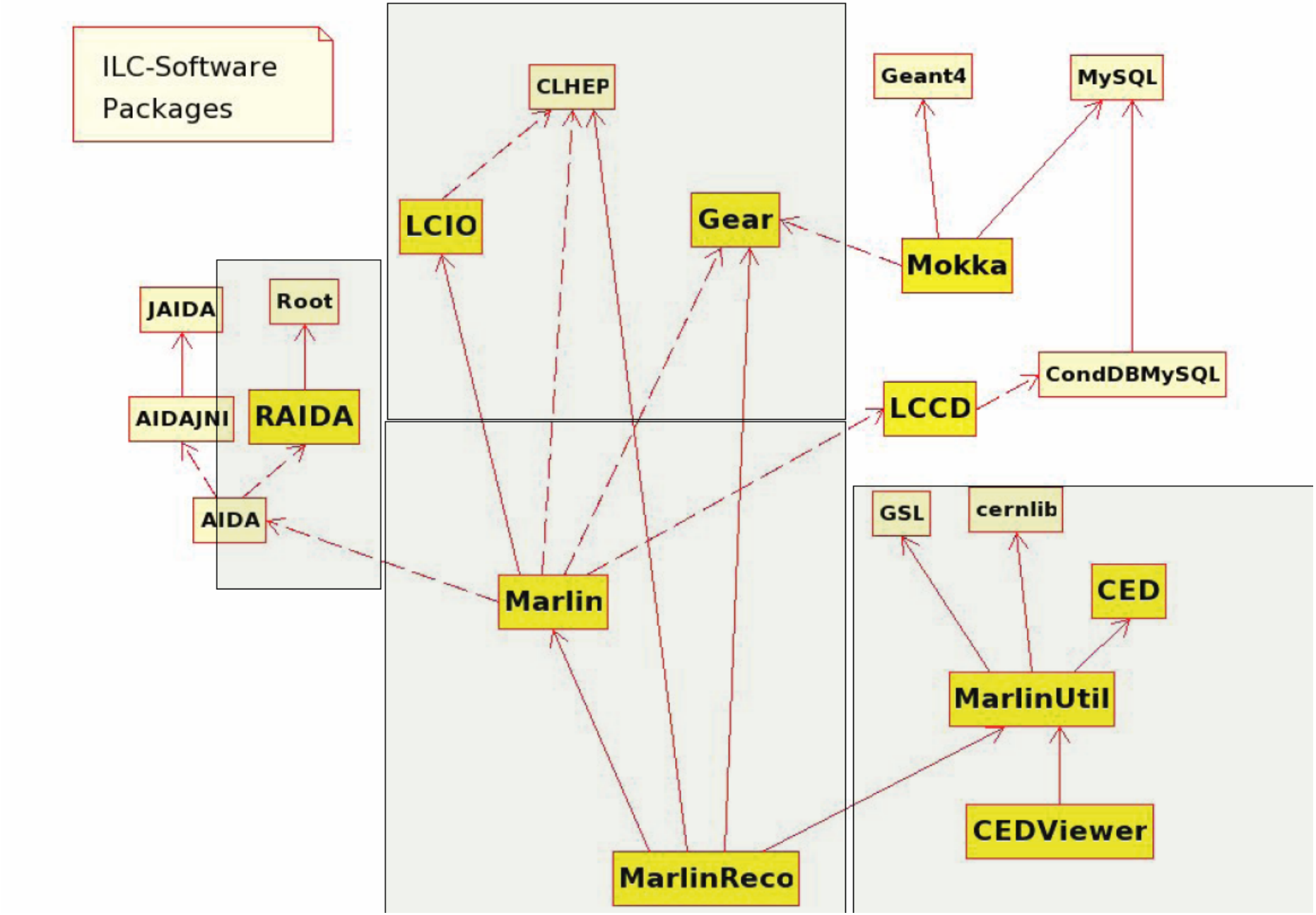
software package dependencies



example: ILC full simulation



example:ILC full reconstruction



A good starting point :

http://ilcsoft.desy.de/portal/software_packages

Brahms
CED
CEDViewer
Gear
LCCD
LCIO
ilcinstall
Marlin
MarlinReco
MarlinUtil
Mokka
RAIDA

CLHEP
CMake
GSL - GNU Scientific Library
geant4
root

Example processor DumpLCIO (header file)

```
//////////////////////////////////////////////////////////////////////////////
//
// Example marlin processor to dump the content of an lcio file
//
//////////////////////////////////////////////////////////////////////////////

#ifndef DumpLCIO_h
#define DumpLCIO_h 1

#include "marlin/Processor.h"
#include "lcio.h"
#include <string>

using namespace lcio;
using namespace marlin;


class DumpLCIO : public Processor
{

   public:

   virtual Processor*  newProcessor() {return new DumpLCIO;}

   /** Constructor
   * Register here the steering input parameters
   */
   DumpLCIO();


   /** Called at the beginning of the job before anything is read.
   * Use to initialize the processor, e.g. book histograms.
   */
   virtual void init();


   /** Called for every run.
   */
   virtual void processRunHeader(LCRunHeader* run);


   /** Called for every event - the working horse.
   */
   virtual void processEvent(LCEvent* evt);


   virtual void check(LCEvent* evt);


   /** Called after data processing for clean up.
   */
   virtual void end();


   protected:

   std::string  colName;      //input collection name
   double  anInputParameter;  //an input/steering parameter

   int  nRun;
   int  nEvt;
};

#endif
//////////////////////////////////////////////////////////////////////////////
```

Example processor DumpLCIO (source file)

```
//////////////////////////////////////////////////////////////////////////////

#include "DumpLCIO.h"
#include <iostream>

#include <EVENT/LCCollection.h>
#include <EVENT/MCParticle.h>
#include "EVENT/SimCalorimeterHit.h"

using namespace lcio;
using namespace marlin;

DumpLCIO aDumpLCIO;

//.....................................................................
DumpLCIO::DumpLCIO() : Processor("DumpLCIO")
{
   // modify processor description
    description = "DumpLCIO does this ... and that ...";


   // register steering parameters: name, description, class-variable,
   //                               default value

   registerProcessorParameter("CollectionName",
                              "Name of the collection to process",
                               colName,
                              std::string("MCParticle"));

   registerProcessorParameter("AnInputParameter",
                              "Some input parameter",
                               anInputParameter,
                              double(999.));
}
//.....................................................................
void DumpLCIO::init()
{
   // print steering parameters
   printParameters();

    nRun = 0;
    nEvt = 0;
}
//.....................................................................
void DumpLCIO::processRunHeader(LCRunHeader* run)
{
    nRun++ ;
}
//.....................................................................
void DumpLCIO::processEvent(LCEvent* evt)
{
   // this is called for every event
   // usually the working horse ...

   std::cout << "Event: " <<  nEvt << " " << evt->getEventNumber()
   << " ....................."<< std::endl;

   //loop over the collections of the event
   //
   typedef const std::vector <std::string> StringVec;
   StringVec* strVec = evt->getCollectionNames();
   for(StringVec::const iterator name = strVec->begin();
       name != strVec->end(); name++)
   {
      LCCollection* col = evt->getCollection(*name);

      std::cout << "=> collection,type,elements: "
                << " " << *name
                << " " << col->getNumberOfElements()
                << std::endl;
   }


   //get collection by name (= parameter in the steering file)
   //
   //this is problematic, if the collection does not exist
   //it throws an exception and aborts
   //LCCollection* col = evt->getCollection( colName);
   //
   //a workaround is to loop over the vector of names
   //
   for(StringVec::const iterator name = strVec->begin();
       name != strVec->end(); name++)
   {
      if(*name ==  colName)
      {
         LCCollection* col = evt->getCollection(*name);

         std::cout << "Collection " <<  colName << " has "
         << col->getNumberOfElements() << " elements" << std::endl;

         //loop over the elements of the collection
         //
         if(col->getTypeName() == "SimCalorimeterHit")
         {
            for(int i=0; i < col->getNumberOfElements(); i++)
            {
               SimCalorimeterHit* elem =
               dynamic_cast<SimCalorimeterHit*>(col->getElementAt(i));

               std::cout << "element,id,x,y,z,energy: "
                    << i
                    << " " << elem->getCellID0()
                    << " " << elem->getPosition()[0]
                    << " " << elem->getPosition()[1]
                    << " " << elem->getPosition()[2]
                    << " " << elem->getEnergy()
                    << std::endl;
            }
         }
         else
         {
            std::cout << "Event: " << evt->getEventNumber()
            << " has no " <<  colName << " collection" std::endl;
         }
      }

    nEvt++;
}
//.....................................................................
void DumpLCIO::check(LCEvent* evt)
{
   // nothing to check here - could be used to fill checkplots in
   // reconstruction processor
}
//.....................................................................
void DumpLCIO::end()
{
//   std::cout << "DumpLCIO::end()   " << name()
//          << " processed " <<  nEvt << " events in " <<  nRun << " runs "
//          << std::endl ;
}
//.....................................................................
```

Example steering file

```
##########################################################################
#
#  Example steering file for Marlin
#
##########################################################################


### global steering

.begin Global  -------------------------------------------------------------

   # specify one ore more input files (in one ore more lines)
   # LCIOInputFiles ../simjob.slcio ../simjob1.slcio
   #
   LCIOInputFiles /r08/calice/mc/lcio/uds91 01 LHEP.slcio


   # the active processors that are called in the given order
   #
   ActiveProcessors DumpLCIO
   #ActiveProcessors MyAnalysis


   # limit the number of processed records (run+evt):
   #
   MaxRecordNumber 21


   # don't call the check method of the processors if "true"
   #SupressCheck true

.end   ---------------------------------------------------------------------


### steering per processor

.begin DumpLCIO ------------------------------------------------------------
ProcessorType DumpLCIO

   # set steering parameters

   CollectionName   LumiCalS LumiCal
   AnInputParameter 888

.end ------------------------------------------------------------------------
```

Example output

```
---- DumpLCIO -  parameters:
        AnInputParameter:  888
        CollectionName:  LumiCalS LumiCal
------------------------------------------------

Event: 0 0 .....................

=> collection,type,elements:  LumiCalS LumiCal SimCalorimeterHit 15
=> collection,type,elements:  MCParticle MCParticle 240
=> collection,type,elements:  SEcal01 EcalBarrel SimCalorimeterHit 1860
=> collection,type,elements:  SEcal01 EcalEndcap SimCalorimeterHit 287
=> collection,type,elements:  SHcal01 HcalBarrelEnd SimCalorimeterHit 177
=> collection,type,elements:  SHcal01 HcalBarrelReg SimCalorimeterHit 814
=> collection,type,elements:  SHcal01 HcalEndCaps SimCalorimeterHit 124
=> collection,type,elements:  STpc01 FCH SimTrackerHit 15
=> collection,type,elements:  STpc01 TPC SimTrackerHit 33694
=> collection,type,elements:  ftd01 FTD SimTrackerHit 35
=> collection,type,elements:  sit00 SIT SimTrackerHit 124
=> collection,type,elements:  vxd00 VXD SimTrackerHit 158

Collection LumiCalS LumiCal has 15 elements

element,id,x,y,z,energy: 0 1900545 172.188 0.0261799 -3054.75 0.00042261
element,id,x,y,z,energy: 1 2031617 178.438 0.0261799 -3054.75 0.000357882
element,id,x,y,z,energy: 2 5378 81.5625 1.12574 -3060.75 0.000711599
element,id,x,y,z,energy: 3 23298 81.5625 4.79093 -3060.75 0.00020501
element,id,x,y,z,energy: 4 1769475 165.938 0.0261799 -3066.75 0.000401822
element,id,x,y,z,energy: 5 28676 81.5625 5.89049 3072.75 0.000277995
element,id,x,y,z,energy: 6 9 81.5625 0.0261799 3102.75 0.000596608
element,id,x,y,z,energy: 7 65545 84.6875 0.0261799 3102.75 0.000234214
element,id,x,y,z,energy: 8 131081 87.8125 0.0261799 3102.75 0.000180842
element,id,x,y,z,energy: 9 196617 90.9375 0.0261799 3102.75 1.35047e-05
element,id,x,y,z,energy: 10 262153 94.0625 0.0261799 3102.75 0.000338586
element,id,x,y,z,energy: 11 327689 97.1875 0.0261799 3102.75 0.000259722
element,id,x,y,z,energy: 12 3932171 269.062 0.0261799 3114.75 0.000455174
element,id,x,y,z,energy: 13 3997707 272.188 0.0261799 3114.75 0.000617871
element,id,x,y,z,energy: 14 2572 81.5625 0.549779 3120.75 0.000123065
```

The equivalent main() code

```
//>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
// date  : 050615
// author: G.Mavromanolakis
//
// code  : read lcio file and dump its collections to screen
//
//<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<


#include <iostream>
#include <iomanip>

#include "lcio.h"
#include "IO/LCReader.h"
#include "IMPL/LCTOOLS.h"
#include "EVENT/LCRunHeader.h"
#include "EVENT/LCCollection.h"
#include "EVENT/SimCalorimeterHit.h"
#include "EVENT/CalorimeterHit.h"
#include "EVENT/MCParticle.h"
#include "EVENT/LCIntVec.h"
#include "EVENT/TrackerHit.h"

using namespace std;
using namespace lcio;

//////////////////////////////////////////////////////////////////////////////////
int main(int argc, char** argv )
{
   char* FileName;
   int RunNumber ;
   int EventNumStart;
   int EventNumEnd;


   // read file name from command line
   if( argc < 5 )
   {  cout << "ERROR --- BAD USAGE" << endl
           << "Usage  : dumplcio <File>  <Run#>  <EventStart> <End> " << endl
           << "Example: dumplcio ./dirpath/Run100121.slcio 100121 0 10"<< endl;

      return 1;
   }

   FileName      = argv[1];
   RunNumber     = atoi(argv[2]);
   EventNumStart = atoi(argv[3]);
   EventNumEnd   = atoi(argv[4]);


   LCReader* lcReader = LCFactory::getInstance()->createLCReader();
   LCEvent* evt(0);
   lcReader->open(FileName);

   LCRunHeader* runHdr = lcReader->readNextRunHeader();
   cout << "RunNumber = " << runHdr->getRunNumber() << endl;

   //loop over events
   for(int nev=EventNumStart; nev<=EventNumEnd; nev++)
   {
      int evtNumber = nev;
      cout << "EventNumber " << evtNumber << endl;
      evt = lcReader->readEvent(RunNumber, evtNumber) ;
      if(!evt)
      {  cout << " couldn't find event " << evtNumber << " - run " << RunNumber
         << " in file " << FileName << endl ;
         continue;
      }

      //loop over collections
      typedef const std::vector <std::string> StringVec;
      StringVec* strVec = evt->getCollectionNames();
      for( StringVec::const_iterator name = strVec->begin();
           name != strVec->end(); name++)
      {
         LCCollection* col = evt->getCollection(*name);

         cout << "-> collection,type,elements: "
              << " " << *name
              << " " << col->getTypeName()
              << " " << col->getNumberOfElements()
              << endl;


         if(*name == "LumiCalS LumiCal")
         {
            //loop over hits of collection
            for(int i=0; i < col->getNumberOfElements(); i++)
            {
               SimCalorimeterHit* elem =
               dynamic_cast<SimCalorimeterHit*>(col->getElementAt(i));

               cout << "element,id,x,y,z,energy: "
                    << i
                    << " " << setw(10)  << elem->getCellID0()
                    << " " << setw(10)  << elem->getPosition()[0]
                    << " " << setw(10)  << elem->getPosition()[1]
                    << " " << setw(10)  << elem->getPosition()[2]
                    << " " << setw(10)  << elem->getEnergy()
                    << endl;

            }
         }

      }//end loop over collections
   }//end loop over events
   lcReader->close() ;


   return 0 ;
}
//////////////////////////////////////////////////////////////////////////////////
```