

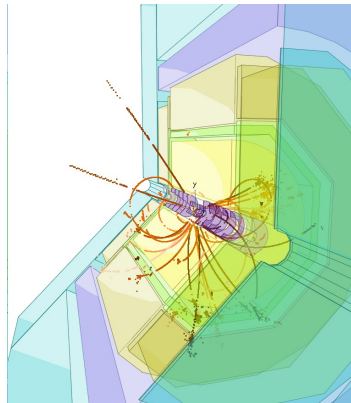
LCIO

Brief History, Status and Plans

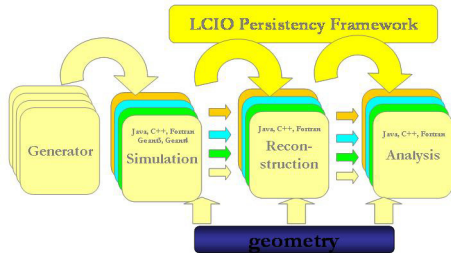
F.Gaede, DESY

EDM4Hep Discussion, Jul 9, 2019

- Brief History
- Overview
- Main Features
- Some Lessons learned
- Future plans
- Towards EDM4Hep
- Summary

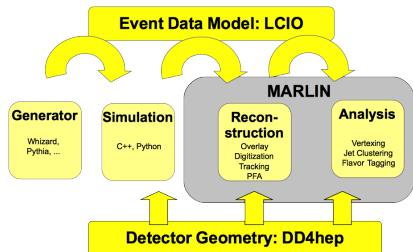


- in 2002 there were three LC projects in the world: Tesla, JLC and NLC
 - and about four or five different detector concepts and software frameworks
 - using C++, Java and Fortran (no Python yet)
- decided to provide the basis for collaboration and common development by defining the common language, i.e. the event data model: **LCIO**
- adding **Marlin** application framework already provided the basis for iLCSoft
- last major evolution: developed and incorporated the **DD4hep** geometry toolkit



from CHEP2003 LCIO paper

- in 2002 there were three LC projects in the world: Tesla, JLC and NLC
 - and about four or five different detector concepts and software frameworks
 - using C++, Java and F77 (no Python yet)
- decided to provide the basis for collaboration and common development by defining the common language, i.e. the event data model: **LCIO**
- adding **Marlin** application framework already provided the basis for iLCSoft
- last major evolution: developed and incorporated the **DD4hep** geometry toolkit



iLCsoft schema today

- started in 2002 as a joint DESY, SLAC and LLR project
- first introduced at CHEP 2003 in San Diego

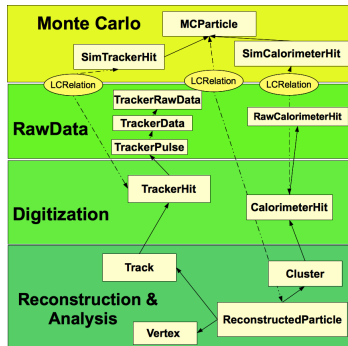
- initially only **Simulation EDM**:
 - *MCParticle, SimCalorimeterHit, SimTrackerHit*
- later extended with **Reconstruction EDM**
 - *CalorimeterHit, TrackerHit, Track, Cluster, ReconstructedParticle,...*

- updated the EDM over the years, as needed/requested by the LC-community
 - have to foresee this kind of *schema evolution*

users of LCIO

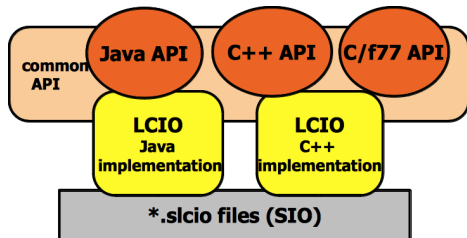
- ILD, SiD, CLICdp, CEPC, FCC-ee (partly), Calice, LC-TPC, EU-Telescope, HPS,...

- LCIO provides the common **EDM** and **persistency**
 - originally support Java, C++ and F77 - now effectively only C++ used
- hierarchical EDM, objects stored in named collections, implementation strictly separated from persistency:
- **SIO** a simple binary I/O system with
 - compression
 - machine agnostic format
 - pointer chasing



<http://lcio.desy.de>

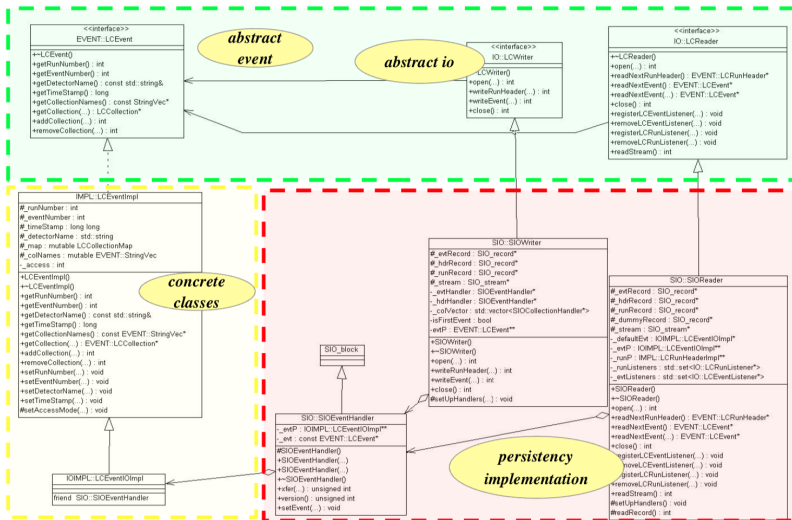
- two independent implementations for Java and C++
- auto generated interface files
 - from *.aid files
 - with Java/C++ like syntax
- the actual *SIO* streaming code is written *manually*
- C interface developed as wrappers to C++ implementation
 - used w/ CFOTRAN.h for F77 API



- a simple binary I/O system with
- zlib compression - per (event) record
- machine agnostic via XDR
 - uses BigEndian
 - (Somewhat for today's common hardware)
- pointer chasing for object pointers
 - implemented via lookup in maps of *pointedAt* and *pointerTo* IDs
- I/O performance **comparable to ROOT I/O for objects**

- currently SIO is re-implemented for multi-threading ...

- *abstract interface* for reading
- *implementation classes* for writing
- *streaming code* hidden from user code



```
LCWriter* lcWrt = LCFactory::getInstance()->createLCWriter() ;
lcWrt->open( FILEN , LCIO::WRITE_NEW ) ;

for(int i=0;i<NEVENT;i++){
    LCEventImpl* evt = new LCEventImpl() ;
    evt->setRunNumber( rn ) ;
    evt->setEventNumber( i ) ;
    //...
    lcWrt->writeEvent( evt ) ;
}
lcWrt->close() ;
delete lcWrt ;
```

```
LCReader* lcReader = LCFactory::getInstance()->createLCReader(
    LCReader::directAccess) ;

lcReader->open( FILEN ) ;

LCEvent* evt=lcReader->readEvent(runNumber, evtNumber) ;

LCCollection* col = evt->getCollection("MCParticle");

for(int i=0,N=col->getNumberOfElements();i<N;++i){
    MCParticle* mcp = static_cast<MCParticle*>( col->getElementAt(i) );
    std::cout << " particle type : " << mcp->getPDG() << std::endl ;
}
```

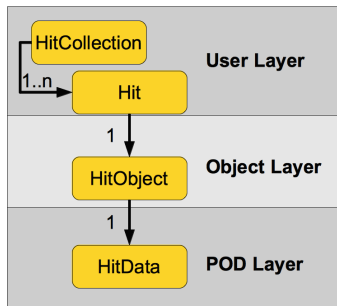
- direct access to individual events
 - implemented with linked list of TOC-records
- streaming mode with *callbacks* for `processEvent/Run()`
- most classes have useful convenient methods, e.g.
 - `MCParticle::isDecayedInTracker()`
- arbitrary relations between objects: 1-to-1, 1-to-N, N-to-M
- optional ROOT dictionary for
 - streaming of LCIO files in ROOT directly
 - provides basis for **Python API**
- many utility classes for:
- dumping of event, collections, elements
 - also with operator `<<()`
- encoding/decoding cellIDs (bitfields) from naming string:
"system:5,module:3,..."
- relation navigator (to-and from-relations) using maps
- `ParticleIDHandler`: access to PID information
- *many more ...*

- defining a common EDM for all linear collider studies provided the basis for developing the common software eco-system *iLCSoft*
- IMHO: a common EDM is an **indispensable ingredient to a turnkey software stack**
- defining an EDM is not entirely *trivial*
 - started out with experienced people from several different experiments (SLC, LEP, HERA)
 - close communication with detector and analysis physicists
 - LCIO EDM was iteratively extended and *improved* over the last **15 years**
- the current LCIO EDM has been battle-proven for many large Monte Carlo campaigns for future e^+e^- colliders: ILC, CLIC, CEPC
- the LCIO EDM clearly fulfills all the needs of the e^+e^- community
 - potentially also usable to a large extend for *hadrons* !?

- the exact details of the EDM don't really matter, as long as users can store and access all information they need for their analyses
- LCIO has been extremely stable over the last decade or so
- the API looks a bit *old fashioned* these days
 - using bare pointers, abstract interfaces, get/set-syntax, . . .
- performance was not one of the main design goals for LCIO
 - there could be room for improvement there

- currently moving to a thread-safe *re-implementation* of SIO in context of development for *MarlinMT*
- plan to modernize the LCIO I/O implementation by using PODIO
- re-implemented the (almost) complete EDM in package *pLCIO*
 - done in AIDA2020 project as *proof-of-concept*
- could use the opportunity to also modernize the EDM itself:
 - consistently introduce components for 3-and 4-vectors
 - use *value-semantics*
 - some minor polishing of EDM-API
 - ...

- PODIO is a new EDM toolkit developed in AIDA2020
- based on the use of **PODs** for the event data objects (**P**lain-**O**ld-**D**ata object)
- PODIO originally developed in context of the **FCC** study
 - addressing the problem in a generic way
 - allowing potential re-use by other HEP groups
 - planned application to **LC** (LCIO) - see next slide



- **pLCIO**: package that implements almost *complete LCIO EDM*:
- original idea to be able to create classes that are almost 100% backward compatible did not fully work out
 - true for most of the actual member functions of the EDM classes
 - not true for handling of collections and collection types, creation of objects, user defined parameters, ...
- planned transition from LCIO to pLCIO would be feasible at '*reasonable cost*'
- potentially we could use this transition to *evolve the LCIO EDM*

pLCIO

- **LCIO** is the EDM and persistency solution for **iLCSoft**, the *software ecosystem* used for linear collider detector studies and beyond
- LCIO is used by: ILD, SiD, CLICdp, CEPC, FCC-ee (partly), Calice, LC-TPC, EU-Telescope, HPS, . . .
- the LCIO EDM is *battle-proven* in many e^+e^- -studies over the last 15 years
- already started to *modernize* LCIO:
 - move to a thread-safe version of SIO
 - implement *pLCIO* with PODIO
- also the API could use a *face-lift*

connection with EDM4Hep

- LCIO-EDM provides the ideal starting point for EDM4Hep as it is known to be complete
- should have a look at some newer idea in FCC-EDM
 - e.g. LorentzVector, Point, BareParticle, . . .

- web page
 - <http://lcio.desy.de/>
- doxygen documentation
 - http://lcio.desy.de/v02-09/doc/doxygen_api/html/index.html
- Github page
 - <https://github.com/iLCSoft/LCIO>
- users manual (somewhat outdated)
 - http://lcio.desy.de/v02-09/doc/manual_html/manual.html
- reference manual
 - <http://lcio.desy.de/v02-09/doc/lciorefman.ps>

extra material - detailed LCIO EDM

```
# -- Vector3D with floats
plcio::FloatThree :
  x : float
  y : float
  z : float
ExtraCode :
  ...

# -- Vector3D with doubles
plcio::DoubleThree :
  x : double
  y : double
  z : double
ExtraCode :
  declaration: "
  DoubleThree() : x(0),y(0),z(0) {}\n
  DoubleThree(const double* v) : x(v[0]),y(v[1]),z(v[2]) {}\n
  DoubleThree(const float* v) : x(v[0]),y(v[1]),z(v[2]) {}\n
  double operator[](unsigned i) const { return *( &x + i ) ; }\n
  "

# -- Vector2D with ints
plcio::IntTwo :
  a : int
  b : int
ExtraCode :
  ...
```

```
#----- LCIO TrackState
plcio::TrackState:
  location      : int
  D0            : float
  phi          : float
  omega        : float
  Z0           : float
  tanLambda    : float
  referencePoint : plcio::FloatThree
  covMatrix     : std::array<float,15>
ExtraCode :
  ...

#----- ObjectID helper struct for references/relations
plcio::ObjectID:
  index        : int
  collectionID : int
ExtraCode :
  ...
```

```
plcio::MCParticle:
```

```
Description: "The Monte Carlo particle."
```

```
Author : "F.Gaede, DESY"
```

```
Members:
```

```
- int PDG //PDG code of the particle
- int generatorStatus //status of the particle as defined by the generator
- int simulatorStatus //status of the particle from the simulation program
- float charge //particle charge
- float time //creation time of the particle in [ns] wrt. the event, e.g. for preassigned decays or decays in flight
- double mass //mass of the particle in [GeV]
- plcio::DoubleThree vertex //production vertex of the particle in [mm].
- plcio::DoubleThree endpoint //endpoint of the particle in [mm]
- plcio::FloatThree momentum //particle 3-momentum at the production vertex in [GeV]
- plcio::FloatThree momentumAtEndpoint //particle 3-momentum at the endpoint in [GeV]
- plcio::FloatThree spin //spin (helicity) vector of the particle.
- plcio::IntTwo colorFlow //color flow as defined by the generator
```

```
OneToManyRelations:
```

```
- plcio::MCParticle parents // The parents of this particle.
- plcio::MCParticle daughters // The daughters this particle.
```

```
#----- LCIO LCRunHeader
```

```
plcio::LCRunHeader:
```

```
Description: "Interface for the run header."
```

```
Author : "F.Gaede, DESY"
```

```
Members:
```

- int runNumber //run number
- std::string detectorName //name of the detector setup used in the simulation.
- std::string description //description of the simulation, physics channels etc.

```
VectorMembers:
```

- std::string activeSubdetectors //active subdetectors used in the simulation.

```
#----- LCIO EventHeader
```

```
plcio::EventHeader:
```

```
Description: "Meta information on the event header information - including collection names and types"
```

```
Author : "F.Gaede"
```

```
Members:
```

- int eventNumber //event number
- int runNumber //run number
- long timeStamp //time stamp
- std::string detectorName //detector model

```
VectorMembers:
```

- std::string collectionNames //collection names
- std::string collectionTypes //collection Types

```
#----- LCIO SimTrackerHit
```

```
plcio::SimTrackerHit:
```

```
Description: "LCIO simulated tracker hit"
```

```
Author : "F.Gaede, DESY"
```

```
Members:
```

- int cellID0 //ID of the sensor that created this hit
- int cellID1 //second ID of the sensor that created this hit
- float EDep //energy deposited in the hit [GeV].
- float time //proper time of the hit in the lab frame in [ns].
- float pathLength //path length of the particle in the sensitive material that resulted in this hit.
- int quality //quality bit flag.
- plcio::DoubleThree position //the hit position in [mm].
- plcio::FloatThree momentum //the 3-momentum of the particle at the hits position in [GeV]

```
OneToOneRelations:
```

- plcio::MCParticle MCParticle //MCParticle that caused the hit.


```
#----- LCIO CaloHitContribution
```

```
plcio::CaloHitContribution:
```

```
Description: "Monte Carlo contribution to SimCalorimeterHit"
```

```
Author : "F.Gaede, DESY"
```

```
Members:
```

- int PDG //PDG code of the shower particle that caused this contribution.
- float energy //energy in [GeV] of the this contribution
- float time //time in [ns] of this contribution
- plcio::FloatThree stepPosition //position of this energy deposition (step)

```
OneToOneRelations:
```

- plcio::MCParticle particle //primary MCParticle that caused the shower responsible for this contribution to the hit.

```
#----- LCIO SimCalorimeterHit
```

```
plcio::SimCalorimeterHit:
```

```
Description: "LCIO simulated calorimeter hit"
```

```
Author : "F.Gaede, DESY"
```

```
Members:
```

- int cellID0 //detector specific (geometrical) cell id.
- int cellID1 //second detector specific (geometrical) cell id.
- float energy //energy of the hit in [GeV].
- plcio::FloatThree position //position of the hit in world coordinates.

```
OneToManyRelations:
```

- plcio::CaloHitContribution contributions //Monte Carlo step contribution - parallel to particle

```
#----- LCIO LCFloatVec
plcio::LCFloatVec:
  Description: "LCIO LCFloatVec"
  Author : "F.Gaede, DESY"
  VectorMembers:
    - float values /// float values

#----- LCIO LCIntVec
plcio::LCIntVec:
  Description: "LCIO LCIntVec"
  Author : "F.Gaede, DESY"
  VectorMembers:
    - int values /// int values

#----- LCIO LCStrVec
plcio::LCStrVec:
  Description: "LCIO LCStrVec"
  Author : "F.Gaede, DESY"
  VectorMembers:
    - std::string values /// string values
```

```
#----- LCIO RawCalorimeterHit
```

```
plcio::RawCalorimeterHit:
```

```
Description: "LCIO raw calorimeter hit"
```

```
Author : "F.Gaede, DESY"
```

```
Members:
```

- int cellID0 //detector specific (geometrical) cell id.
- int cellID1 //second detector specific (geometrical) cell id.
- int amplitude //amplitude of the hit in ADC counts.
- int timeStamp //time stamp for the hit.

```
#----- LCIO CalorimeterHit
```

```
plcio::CalorimeterHit:
```

```
Description: "LCIO calorimeter hit"
```

```
Author : "F.Gaede, DESY"
```

```
Members:
```

- int cellID0 //detector specific (geometrical) cell id.
- int cellID1 //second detector specific (geometrical) cell id.
- float energy //energy of the hit in [GeV].
- float energyError //error of the hit energy in [GeV].
- float time //time of the hit in [ns].
- plcio::FloatThree position //position of the hit in world coordinates.
- int type //type of hit. Mapping of integer types to names via collection parameters "CalorimeterHitTypeNames" and
- plcio::ObjectID rawHit //reference to RawCalorimeterHit.

```

#----- LCIO TrackerHit
# o FIXME: no specialisation for the different kind of geometries: TrackerHitPlane, TrackerHitZCylindr
# o FIXME: should we define a FloatSix for the covMatrix or use the std::array ???
plcio::TrackerHit:
Description : "LCIO tracker hit"
Author : "F.Gaede, DESY"
Members :
- int cellID0 //ID of the sensor that created this hit
- int cellID1 //second ID of the sensor that created this hit
- int type //type of raw data hit, either one of LCIO::TPCHIT, LCIO::SIMTRACKERHIT - see collection parameters "T
- int quality //quality bit flag of the hit.
- float time //time of the hit.
- float eDep //energy deposited on the hit [GeV].
- float eDepError //error measured on EDep [GeV].
- float edx //dE/dx of the hit in [GeV].
- plcio::DoubleThree position //hit position in [mm].
- std::array<float,6> covMatrix //covariance of the position (x,y,z), stored as lower triangle matrix. i.e. cov(x,x) , cov(y,x) , cov
VectorMembers:
- plcio::ObjectID rawHits //raw data hits. Check getType to get actual data type.

```

```
#----- LCIO LCGenericObject
# o FIXME: this implementation is rather inefficient ...
plcio::LCGenericObject:
  Description: "LCIO LCGenericObject"
  Author : "F.Gaede, DESY"
  Members:
    - int isFixedSize           //true if all objects have a fixed size, i.e getNInt, getNFloat and getNDouble will return values that a
    - std::string typeName      // The type name of the user class (typically the class name)
    - std::string dataDescription // The description string. A comma separated list of pairs of type identifier, one of 'i','f','d' follow
  VectorMembers:
    - int    intVals // Returns the integer value for the given index.
    - float  floatVals // Returns the float value for the given index.
    - double doubleVals // Returns the double value for the given index.
```

```
#----- LCIO Track
```

```
plcio::Track:
```

```
Description: "LCIO reconstructed track"
```

```
Author : "F.Gaede, DESY"
```

```
Members:
```

```
- int type //flagword that defines the type of track.Bits 16-31 are used internally
- float chi2 //Chi^2 of the track fit
- int ndf //number of degrees of freedom of the track fit
- float dEdx //dEdx of the track.
- float dEdxError //error of dEdx.
- float radiusOfInnermostHit //radius of the innermost hit that has been used in the track fit
```

```
VectorMembers:
```

```
- int subDetectorHitNumbers //number of hits in particular subdetectors.Check/set collection variable TrackSubdetectorNames for
- plcio::TrackState trackStates //track states
```

```
OneToManyRelations:
```

```
- plcio::TrackerHit trackerHits //Optionaly ( check/set flag(LCIO::TRBIT_HITS)==1) return the hits that have been used to create t
- plcio::Track tracks //tracks (segementbts) that have been combined to this track
```

#----- LCIO relations

plcio::LCRelation:

Description: "LCIO LCRelation"

Author : "F.Gaede, DESY"

Members:

- plcio::ObjectID from //from-object of the given relation.
- plcio::ObjectID to //to-object of the given relation.
- float weight //weight of the given relation

#----- LCIO reference (pointer)

plcio::LCReference:

Description: "LCIO reference (pointer) to be used for subset collections"

Author : "F.Gaede, DESY"

Members:

- plcio::ObjectID object //object pointed to

```
#---- LCIO ParticleID
```

```
plcio::ParticleID:
```

```
Description: "LCIO ParticleID - in pLCIO these are stored in separate collections"
```

```
Author : "F.Gaede, DESY"
```

```
Members:
```

- int type //userdefined type
- int pDG //PDG code of this id - (999999) if unknown.
- int algorythmType //type of the algorithm/module that created this hypothesis
- float likelihood //likelihood of this hypothesis - in a user defined normalization.

```
VectorMembers:
```

- float parameters //parameters associated with this hypothesis. Check/set collection parameters ParameterNames_PIDAlgorithmTypeNa


```

#----- LCIO cluster
# Changes w.r.t. to original
# o ParticleIDs are now in external collection
plcio::Cluster:
Description: "LCIO cluster"
Author : "F.Gaede, DESY"
Members:
- int          type          //flagword that defines the type of cluster. Bits 16-31 are used internally.
- float        energy        //energy of the cluster [GeV]
- float        energyError   //error on the energy
- plcio::FloatThree position //position of the cluster.
- std::array<float,6> positionError //covariance matrix of the position (6 Parameters)
- float        iTheta        //intrinsic direction of cluster at position Theta. Not to be confused with direction cluster
- float        phi           //intrinsic direction of cluster at position - Phi. Not to be confused with direction cluster
- plcio::FloatThree directionError //covariance matrix of the direction (3 Parameters)
VectorMembers:
- float  shape          //shape parameters - check/set collection parameter ClusterShapeParameters for size and names of param
- float  weight         //weight of a particular cluster
- float  hitContributions //energy contribution of the hits Runs parallel to the CalorimeterHitVec from getCalorimeterHits().
- float  subdetectorEnergies //energy observed in a particular subdetector. Check/set collection parameter ClusterSubdetectorNames
OneToManyRelations:
- plcio::Cluster    clusters    //clusters that have been combined to this cluster.
- plcio::CalorimeterHit hits     //hits that have been combined to this cluster.
- plcio::ParticleID particleIDs //particle IDs (sorted by their likelihood)

## LCIO - reconstructed particle

```

```

#----- LCIO ReconstructedParticle
# Changes w.r.t. to original
F.Gaede, DESY

```