
# LCIO

## A persistency framework for linear collider detector simulation studies

Frank Gaede, DESY, IT

CHEP 2003, San Diego

## People

- Ties Behnke - DESY/SLAC
- Frank Gaede - DESY
- Norman Graf - SLAC
- Tony Johnson - SLAC
- Paulo Mora de Freitas - IN2P3

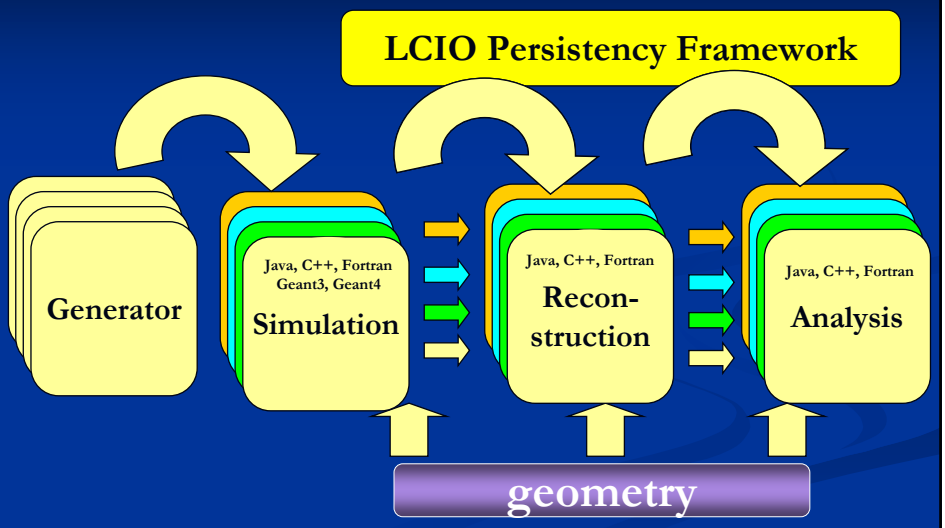
project emerged out of the 'persistency task force' of the ECFA/DESY workshop

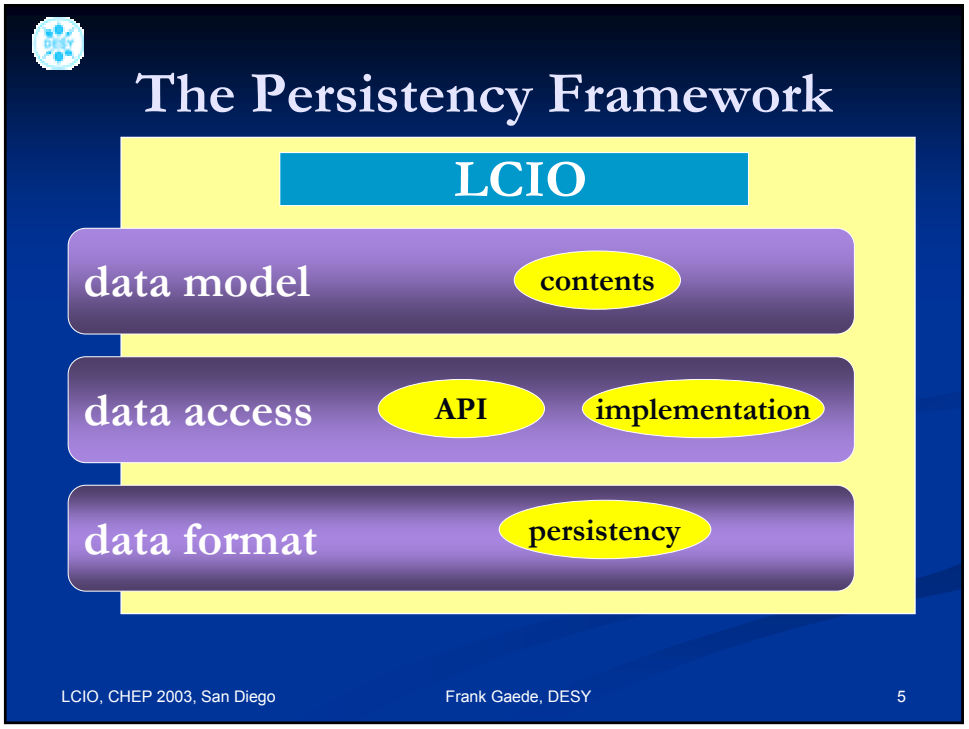
# Outline

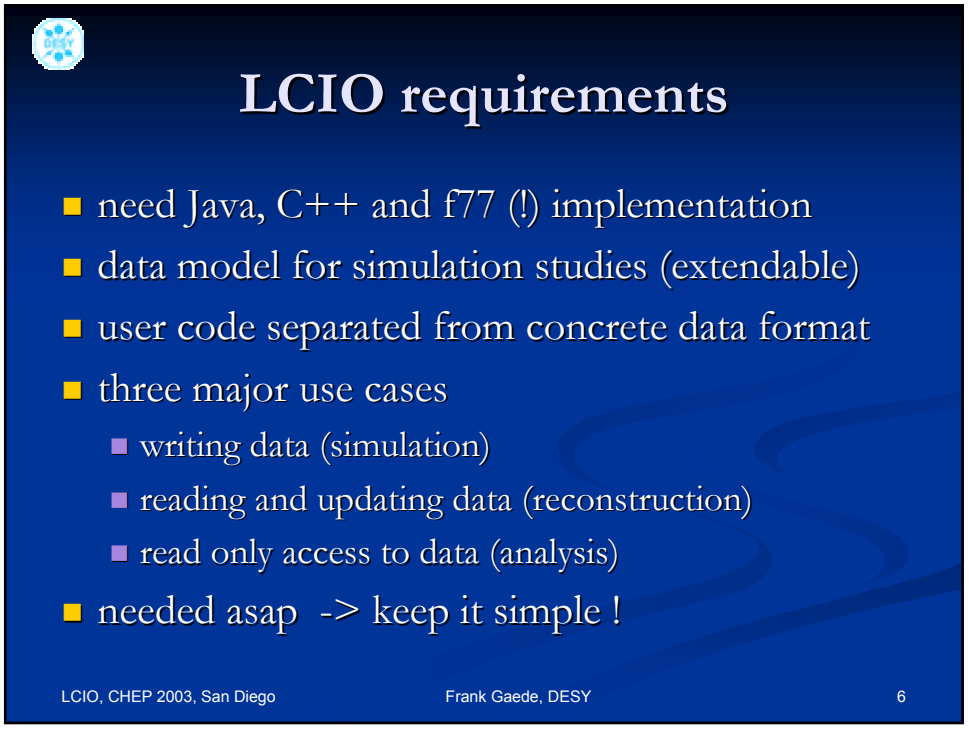
- Introduction
- Data model
- Software design
- Technical realization
- Data format
- Status
- Summary



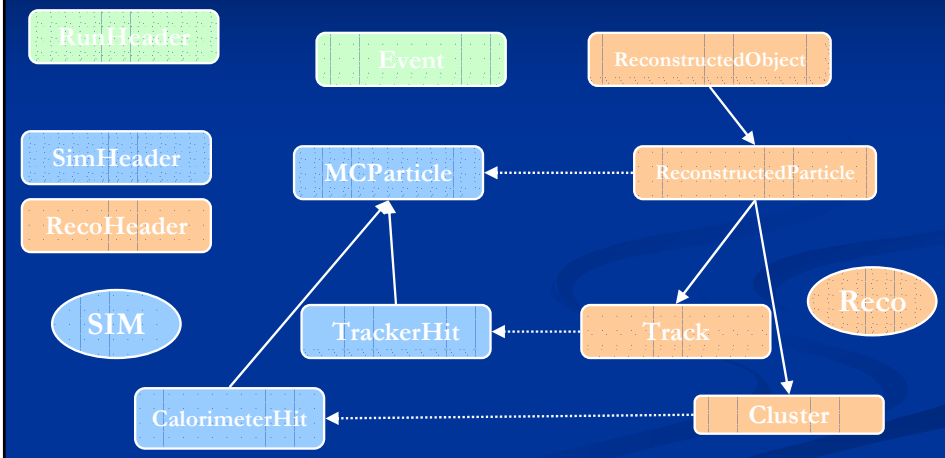
# Motivation




The Persistency Framework
LCIO
data model
contents
data access
API
implementation
data format
persistency
LCIO, CHEP 2003, San Diego
Frank Gaede, DESY
5

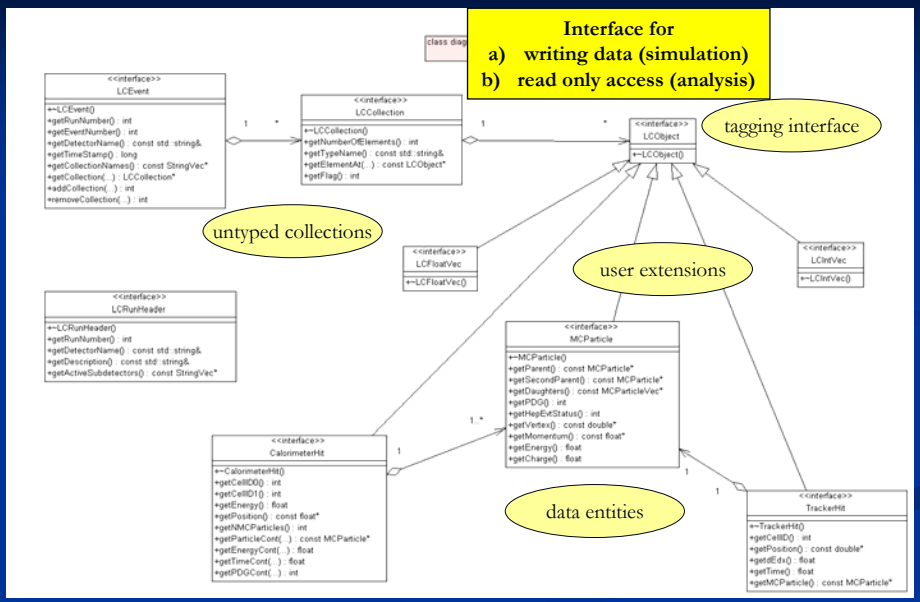

LCIO requirements
need Java, C++ and f77 (!) implementation
data model for simulation studies (extendable)
user code separated from concrete data format
three major use cases
writing data (simulation)
reading and updating data (reconstruction)
read only access to data (analysis)
needed asap -> keep it simple !
LCIO, CHEP 2003, San Diego
Frank Gaede, DESY
6

# A quick Look at the Data Model:

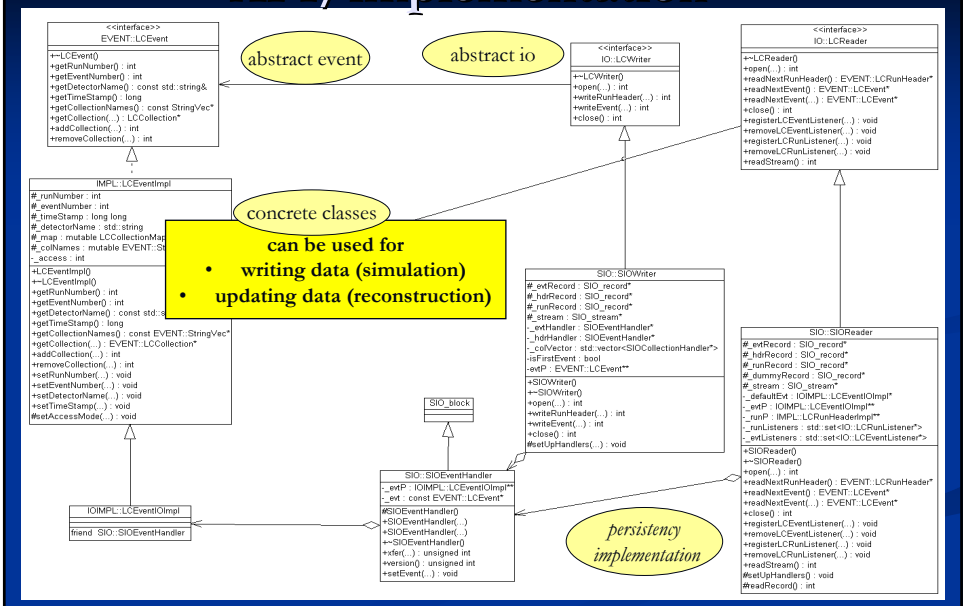





# API – simulation data

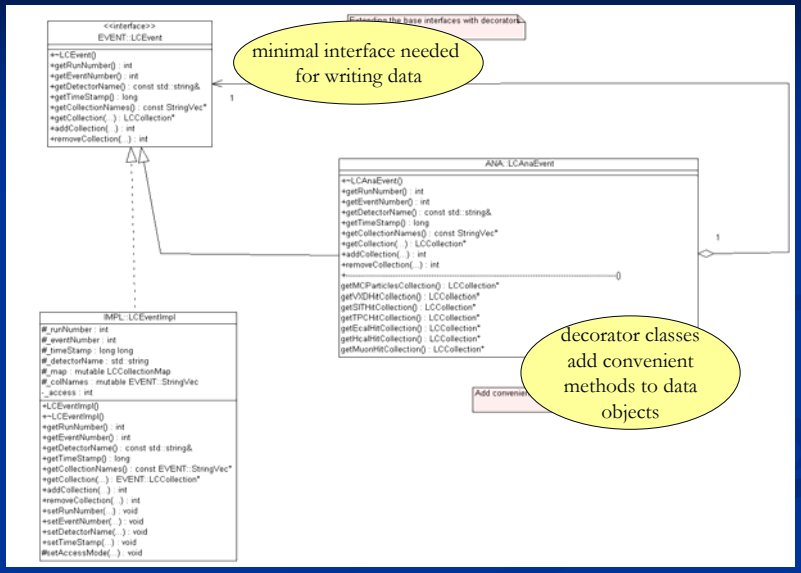


# API/implementation






# Extending the base API

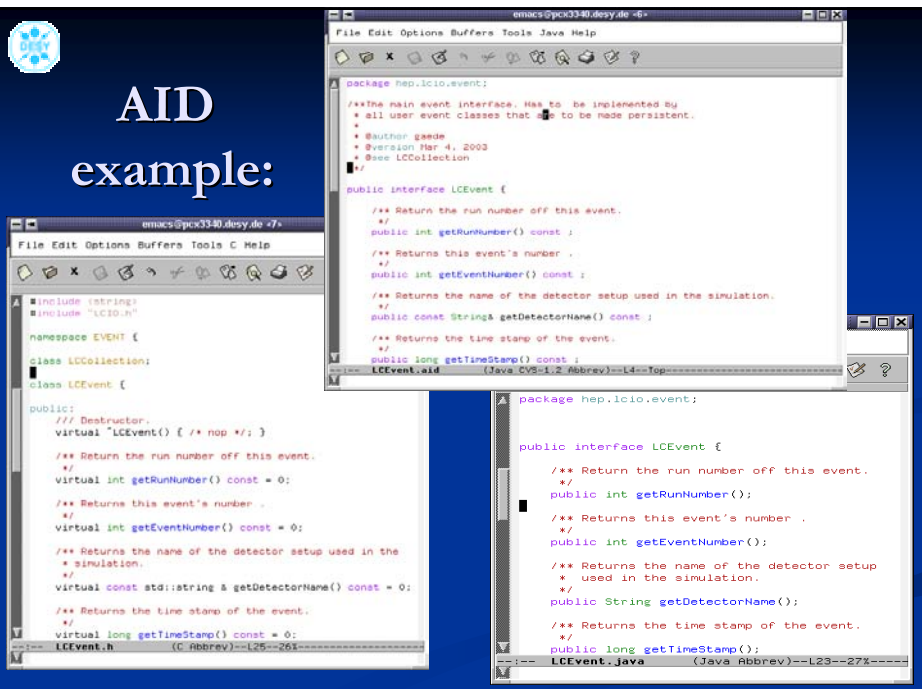


# API definition with AID

- AID Abstract Interface Definition
  - tool from **freehep.org**
  - **used successfully in the AIDA project**
- **define interfaces in Java-like language with C++ extensions**
  - **-> generates files with Java interfaces**
  - **-> generates C++ header files with pure abstract base classes**
- **independent implementations in Java and C++**
  - **-> keep Java "pure" i.e. machine independent**



# AID example:



# LCIO Fortran interface

- Fortran support for
  - legacy software (e.g. BRAHMS reconstruction)
  - non OO-analyses code ("old guys")
- not a third implementation of the library – use C++-wrapper functions and **cfortran.h** instead:
  - one function for every class member function
  - use integers to store pointers !
  - -> OO-like code in fortran



# LCIO f77 example:

emacs@pcx3340.desy.de <6>

File Edit Options Buffers Tools Fortran Help

```
c      ---- event loop ------
       do 10
          event = lrdrreadnextevent( reader )
          if( event.eq.0 ) goto 11

          runnum = levtgetrunnumber( event )
          evtnum = levtgeteventnumber( event )
          detname = levtgetdetectorname( event

          write(*,*) "  run: ", runnum
          write(*,*) "  evt: ", evtnum
          write(*,*) "  det: ", detname

 10    continue
 11    continue
c      ---- end event loop ------
```

--:-- anajob.f (Fortran)--L43--59%----

emacs@pcx3340.desy.de <7>

File Edit Options Buffers Tools C++ Help

```
// ---- event loop ------
const LCEvent*  event ;
while( (event = lcRdr->readNextEvent()) != 0 ){


  int runNum = event->getRunNumber()  ;
  int evtNum = event->getEventNumber()  ;
  string detName = event->getDetectorName() ;

std:cout << "  run: " << runNum << std::endl ;
std:cout << "  evt: " << evtNum << std::endl ;
std:cout << "  det: " << detName << std::endl ;

}

//---- end event loop ------
```

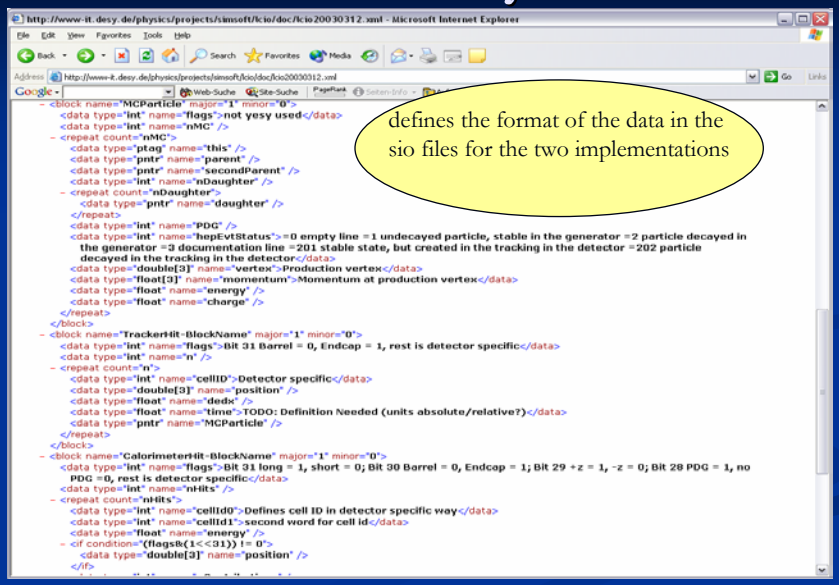
--:-- anajob.cc (C++ Abbrev)--L16-- 4%----------

# Persistency Implementation

- use SIO: Simple Input Output
- developed at SLAC for NLC simulation
- already used in hep.lcd framework
- features:
  - on the fly data compression
  - some OO capabilities, e.g. pointers
  - C++ and Java implementation available
- XML files describing the data layout



# XML data layout:



# Status of LCIO

- C++ implementation available
  - will be integrated into Mokka (geant4) simulation framework
- f77 prototype
  - demonstrating the design
- Java implementation developed now
- complete integration into simulation software chains in the next months:
  - US: hep.lcd (Java)
  - Europe: Mokka (C++)/BRAHMS-reco(f77)



# Summary

- LCIO is a persistency framework for linear collider simulation software
- Java, C++ and f77 user interface
- LCIO is currently implemented in simulation frameworks:
  - hep.lcd
  - Mokka/BRAHMS-reco

  -> other groups are invited to join
- see LCIO homepage for more details:

**http://www-it.desy.de/physics/projects/simsoft/lcio/index.html**