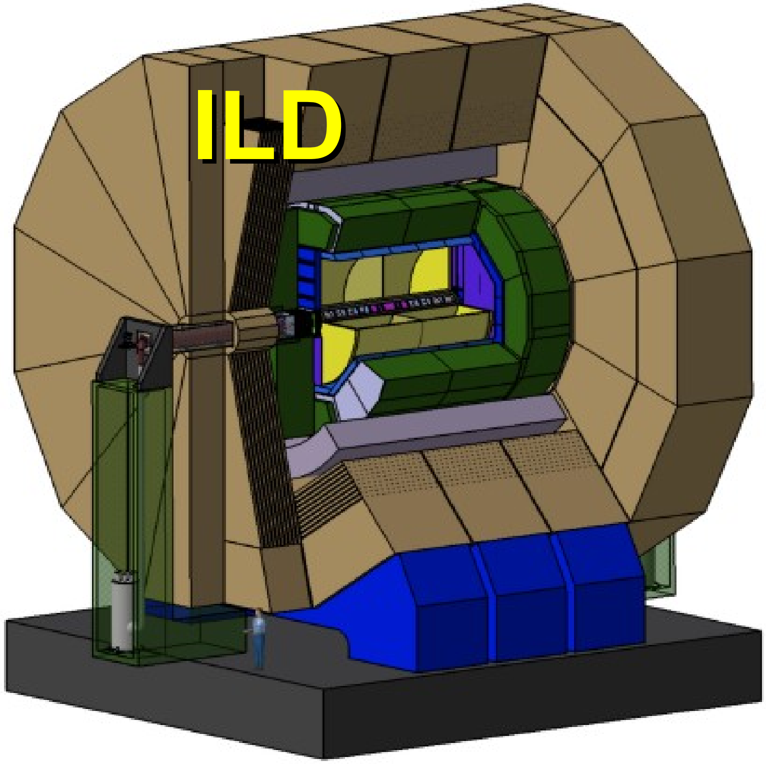
Overview and Plans for Core Software Tools

Frank Gaede
DESY

Linear Collider Testbeam Workshop
LAL Orsay, November 3-5, 2009

Outline

- Introduction
- core tools
  - Mokka, LCIO, Marlin, LCCD,...
- Plans
  - framework tools
  - LCIOv2
- AIDA -fp7
- Grid computing
- Summary

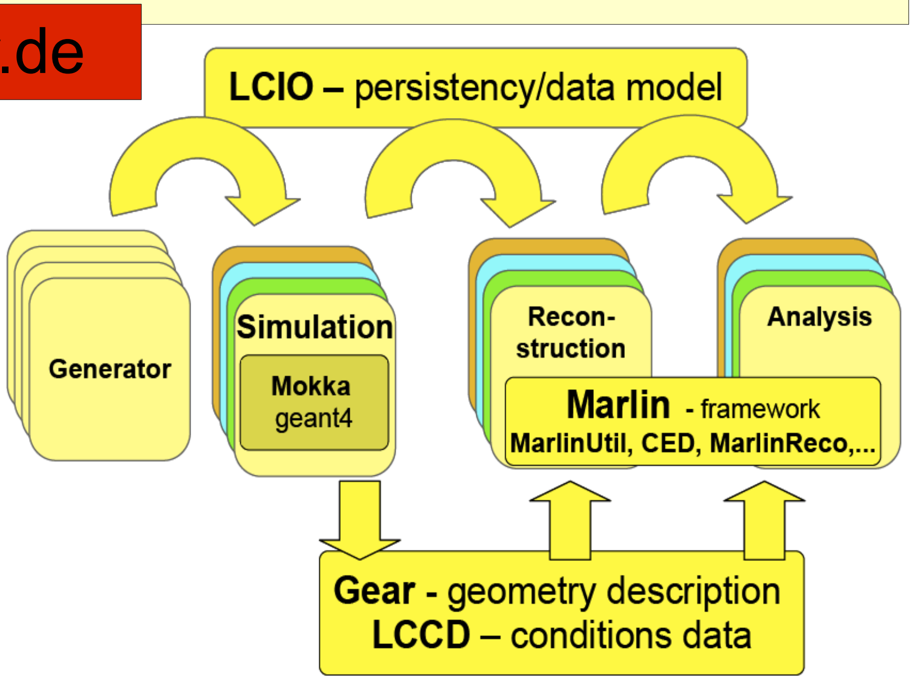


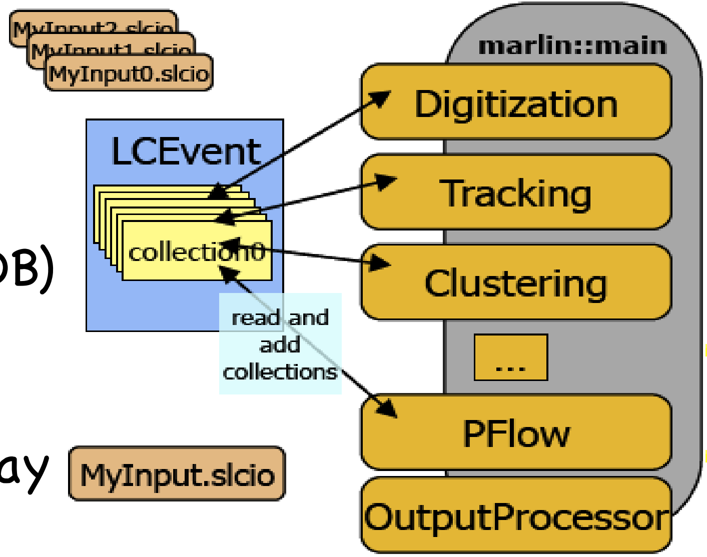
ILD & tbeam Core Software Tools

http://ilcsoft.desy.de

Frank Gaede, DESY, LCTW09, LAL Orsay 3-5 Nov, 2009

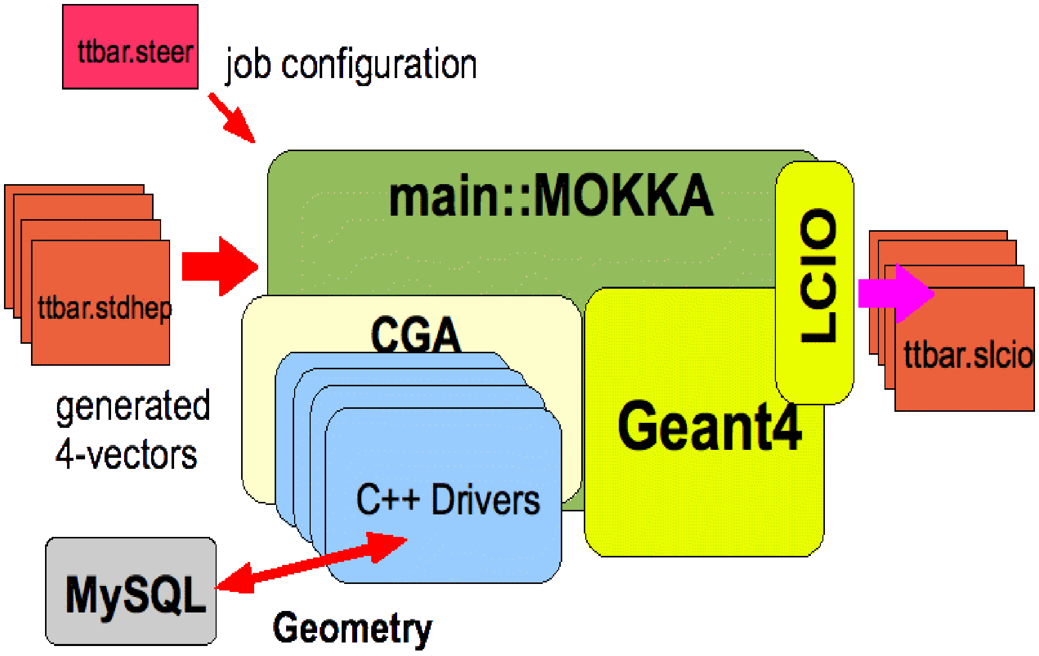
- **Mokka** (LLR)
  - geant4 simulation application
- **LCIO** (DESY/SLAC)
  - international standard for persistency format / event data model
- **Marlin**
  - core application framework for reconstruction & data analysis
- **GEAR** geometry package f. reconstruction
- **LCCD**
  - conditions data toolkit (DB)
- **CED**
  - 3d event display

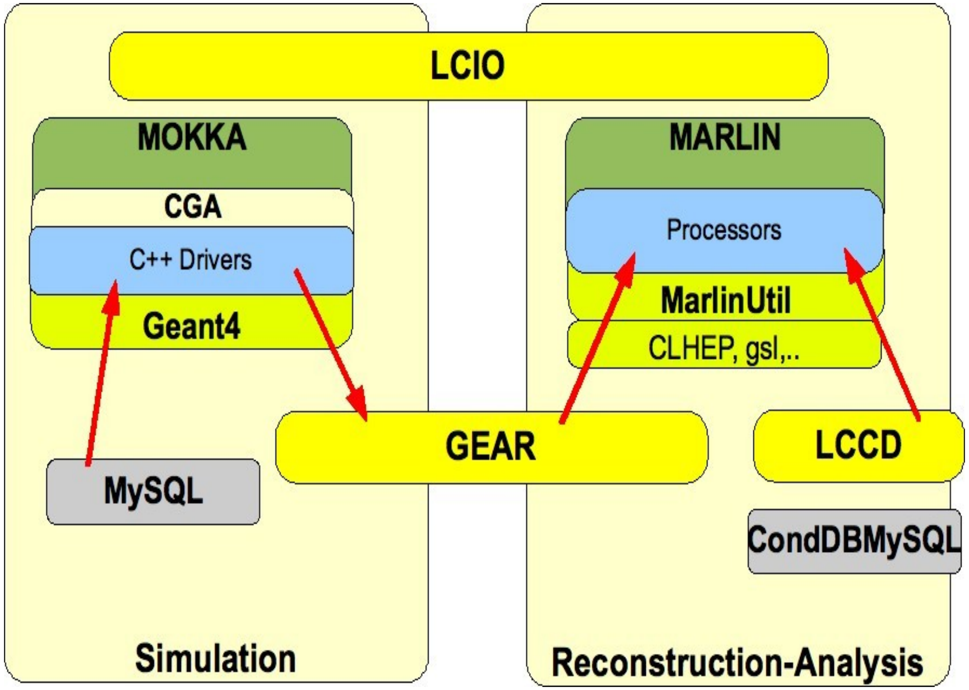




- used for ILD and testbeam:
  - **ILD detector concept** studies
  - **Calice** calo testbeam
  - **LC-TPC** testbeam
  - **EU-PixelTelescope**
- **benefitted from EUDET project**
- **synergies between testbeam and global detector optimization**

Mokka - geant4 simulation

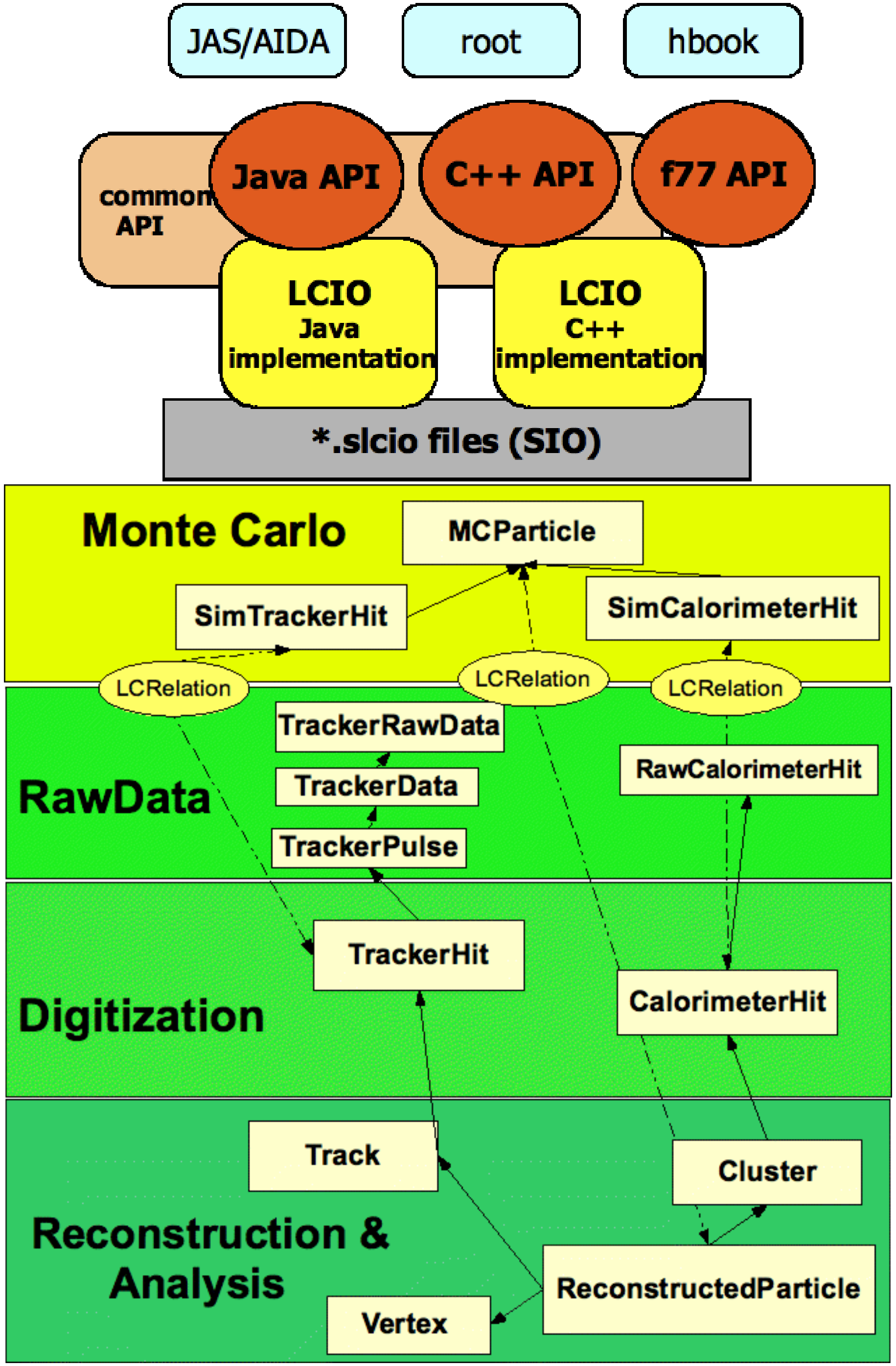




- geant4 based application for simulating the detector response
- flexible geometry on subdetector basis (MySQL & C++ drivers)
  - interface to GEAR geometry API via xml files
- used in ILD detector concept studies
- used in ILC subdetector testbeams: Calice, LCTPC, EUPixelTelescope

LCIO overview

- LCIO provides a hierarchical event data model and a persistency solution for LC detector R&D
- C++ and Java interface – I/O decoupled from user code
- DESY/SLAC project since 2002
- used in ILD and SID SW frameworks and in many ILC **testbeam** experiments

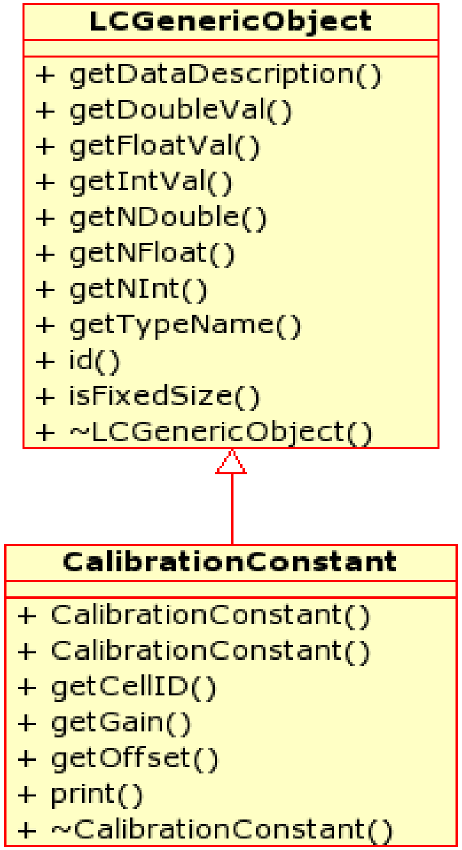


specific raw data classes added in collaboration with tbeam working groups

LCIO - features

- current I/O system SIO
  - Object I/O (w/ pointer chasing)
  - schema evolution
  - compressed records
- mechanism to store user defined classes: LCGenericObject provided by LCIO:
  - store 'arbitrary' data structures in LCGenericObject w/o writing streamer code or dictionaries
  - easy to use - I/O performance not optimal
- LCIO **runtime** extensions (C++)
  - extension of the object with arbitrary (even non-LCObject) classes and relations (no persistency)
- requested features
  - direct access to events
  - splitting of events over files -partial reading of events
  - streamers for arbitrary user classes

... to be addressed in LCIOv2



LCGenericObject

+ getDataDescription()
+ getDoubleVal()
+ getFloatVal()
+ getIntVal()
+ getNDouble()
+ getNFloat()
+ getNInt()
+ getTypeName()
+ id()
+ isFixedSize()
+ ~LCGenericObject()

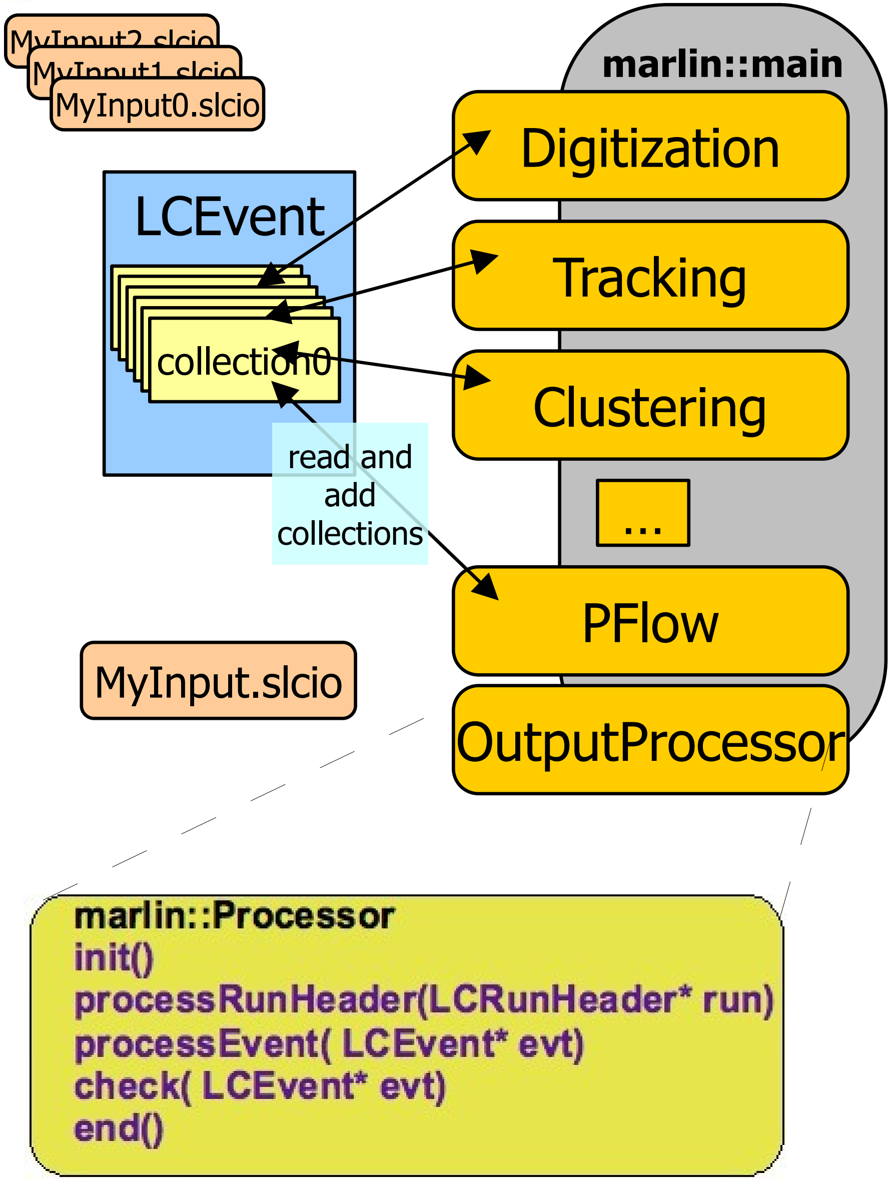
CalibrationConstant

+ CalibrationConstant()
+ CalibrationConstant()
+ getCellID()
+ getGain()
+ getOffset()
+ print()
+ ~CalibrationConstant()

Marlin – core application framework

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

- modular C++ application framework for the analysis and reconstruction of ILC data
- **LCIO** as transient data model: software bus model
- xml steering files:
  - fully configure application
  - parameters global + processor
- self documenting
  - parameters registered in user code
- consistency check of input/output collection types
- **Plug & Play** of modules



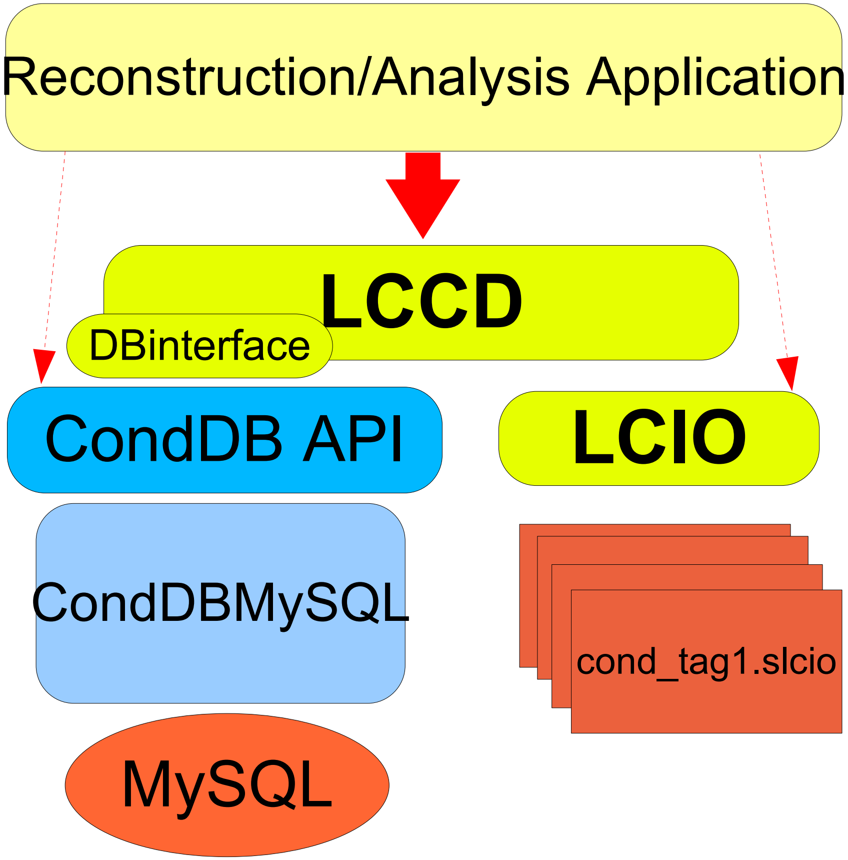
```
marlin::Processor
init()
processRunHeader(LCRunHeader* run)
processEvent( LCEvent* evt)
check( LCEvent* evt)
end()
```

Frank Gaede, DESY, LCTW09, LAL Orsay 3-5 Nov, 2009

LCCD conditions data base

- Reading/Writing conditions data to/from
  - conditions database
  - simple LCIO file
  - LCIO data stream
  - dedicated LCIO DB-snapshot
- fully functional since 2005
- integrated in Marlin
- possible issues ?
  - outphased CondDBMySQL
  - LCGenericObjects streamed in BLOB in MySQL

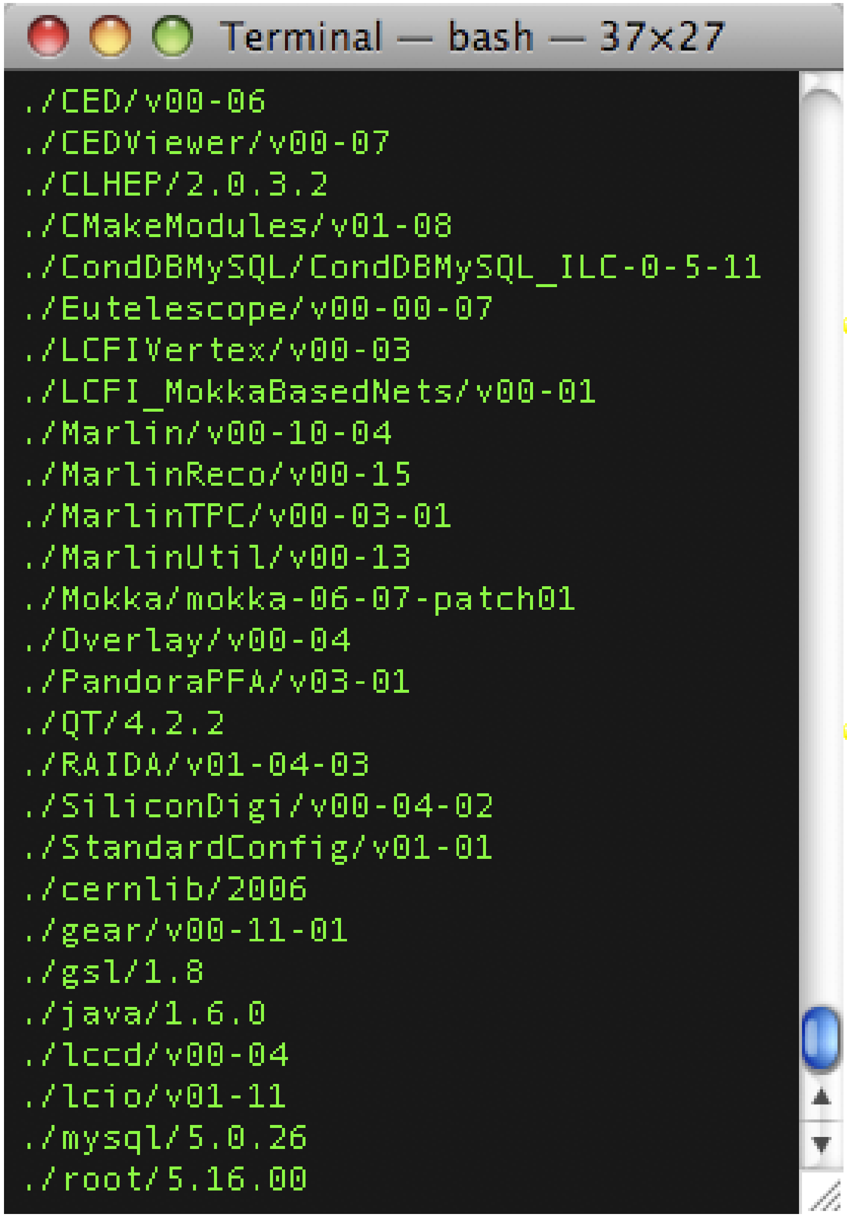
Linear **C**ollider **C**onditions **D**ata Toolkit



LCCD is used for the conditions data of most of the the ongoing ILC testbeam studies

ILD software builds and installation

/afs/desy.de/group/it/ilcsoft/v01-06

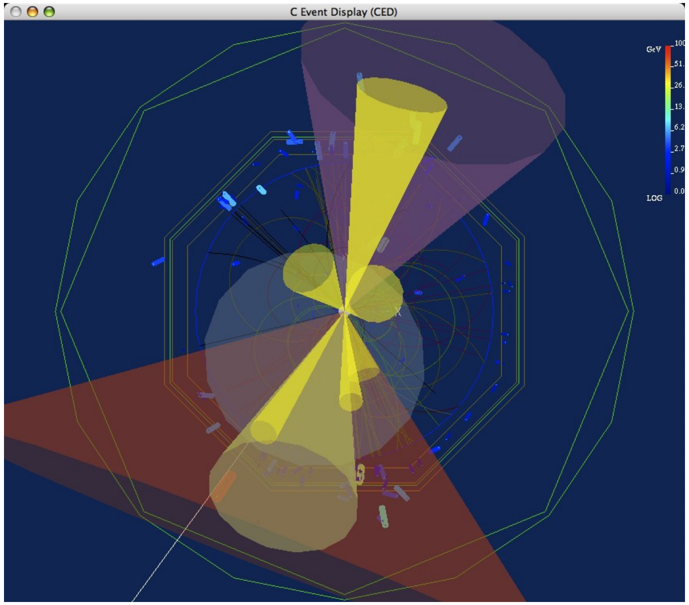


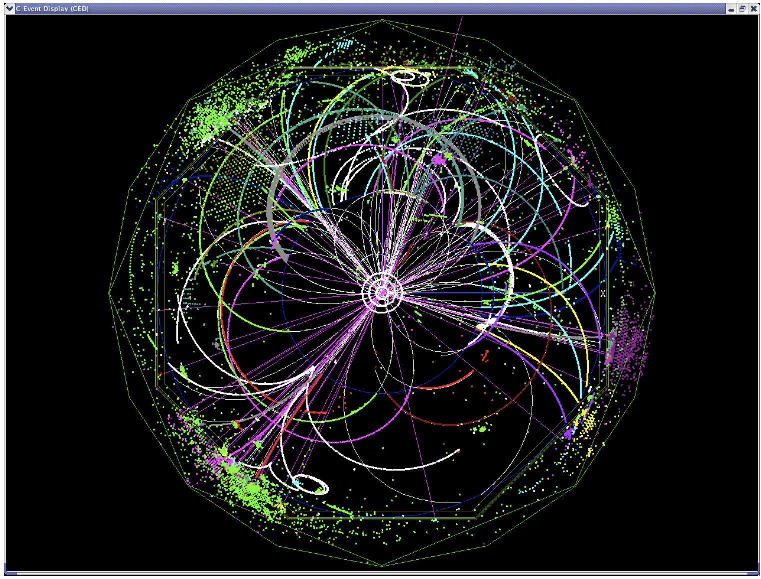
```
./CED/v00-06
./CEDViewer/v00-07
./CLHEP/2.0.3.2
./CMakeModules/v01-08
./CondDBMySQL/CondDBMySQL_ILC-0-5-11
./Eutelescope/v00-00-07
./LCFIVertex/v00-03
./LCFI_MokkaBasedNets/v00-01
./Marlin/v00-10-04
./MarlinReco/v00-15
./MarlinTPC/v00-03-01
./MarlinUtil/v00-13
./Mokka/mokka-06-07-patch01
./Overlay/v00-04
./PandoraPFA/v03-01
./QT/4.2.2
./RAIDA/v01-04-03
./SiliconDigi/v00-04-02
./StandardConfig/v01-01
./cernlib/2006
./gear/v00-11-01
./gsl/1.8
./java/1.6.0
./lccd/v00-04
./lcio/v01-11
./mysql/5.0.26
./root/5.16.00
```

- **ilcinstall** tool: python scripts to download, build and install all ILD core software tools and testbeam packages
- 'edit and start configure script – go to lunch – run ILD software'
  - on 'scratch' disk – provided geant4, root and mysql are installed
- used for
  - **reference installations** in afs (SL4/5)
  - **grid installations** (all WLCG sites supporting VO ILC)
  - **binary tar-balls** (SL4/5)

CED event display

- CED, client server OpenGL event display displays LCIO and is fully integrated in Marlin
- features:
  - fast 3D transformations (real time)
  - display of detector geometry (GEAR), hits, clusters, tracks, RecoParticles (Jets)
  - fish eye view
  - picking (under development)
- most (all) testbeams have their own event displays and don't use CED [missed oportunity for collaboration !?]





Plans for (ILD) core software

- after LOI we need to further improve ILD software and get ready for TDR phase (2012)
- most if this will of course be beneficial for the testbeams as well !
- plans:
  - improve the simulation & reconstruction
  - develop a test system
  - develop new GRID production system
  - improve the geometry description
  - develop LCIOv2

Test system for ILD software

- develop test system for ILD software including:
- unit tests
  - 'technical' software tests on class/function level
- integration tests
  - technical tests of packages and their interplay
- physics quality
  - check algorithms, physics performances, hit maps,...
- need to be pragmatic about this: look into existing testing tools and/or extend our installation toolkit
- should be useful for testbeam software frameworks as well

new GRID production system

- during LOI Monte Carlo production realized that current system needed quite some manual interference and 'baby sitting'
- in order to save manpower with next major production started development of new GRID production system:
  - properly design data base schema (performance)
  - based on python scripts (flexibility & maintainability)
  - better robustness and error handling
  - easy to use (share work of production)
- interest from CLIC to collaborate

Extend and Improve ILD Simulation

- need 'ILD baseline detector' in simulation, with
  - proven subdetector technology
  - including realistic description of 'faults and imperfections'
- need to develop additional technology subdetector drivers for Mokka, such as:
  - SciEcal and DHCal options (ongoing)
  - FPCCD vertex detector
- need to improve realism for some subdetectors wrt. LOI model ILD_00, eg.
  - silicon trackers: SIT, SET, ETD, FTD (currently cylinder and disks w/ parameterized support material)

need simulation contact from detector R&D groups !

develop a generic geometry Toolkit

- description of complex shapes, materials and sensitive detectors
  - with interfaces to:
  - full simulation programs (geant4, fluka?)
  - fast simulation programs
  - reconstruction algorithms
    - high level interface a la GEAR
    - questions that need to be answered during reconstruction tracking and clustering/PFA

  - visualization tools (ROOT, VRML, etc.)
- allow for misalignment of detector components
- small memory footprint
- efficient tracking in geometry hierarchy and fields
- include needs for testbeam setups

ideally collaborate with other HEP groups on that !

Improve Reconstruction Tools

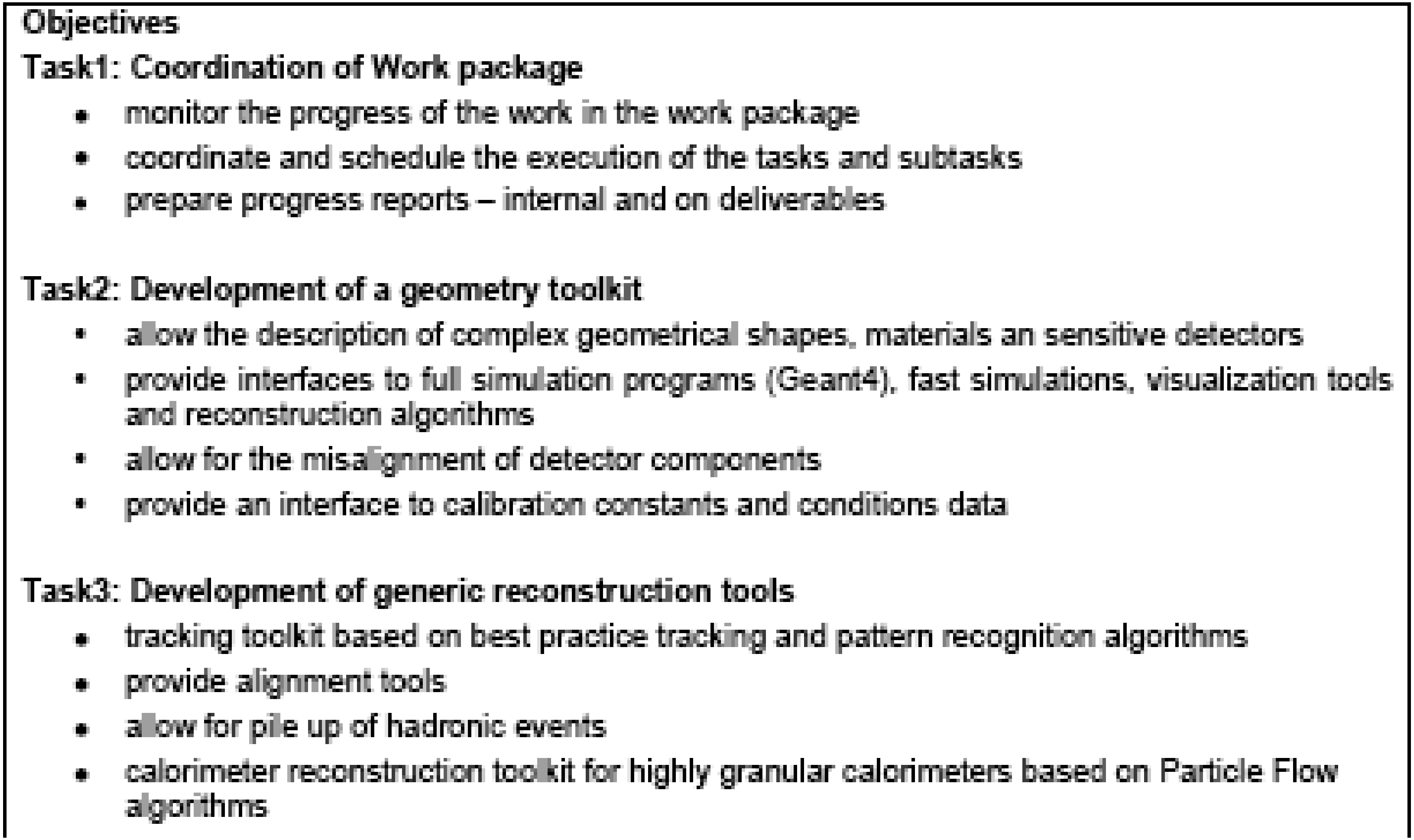
- digitization:
  - improve description of spacial resolution -> feedback from R&D groups needed
  - introduce ghost hits for strip detectors
- tracking:
  - develop modern tracking and pattern recognition software to replace f77 LEPTracking
    - need code for proper treatment of strip detectors
    - ghost hits from stereo layers,...
    - tracking in non-uniform B field (anti-DID)
  - collaborate with tracking R&D groups !

LCIOv2

- improve event data model
  - 1d, 2d hits
  - Track class - multiple fits per track
- Improve I/O
  - splitting of files - partial reading of events
  - direct access
  - srteaming of user classes
- investigate the use of ROOT with LCIO
  - LCEvent in ROOT macros
  - look into optional ROOT I/O for LCIO
- optionally improve existing SIO
  - tens of TByte of testbeam data exist in .SLCIO
  - need to fully support I/O for this data in the future !

fp7 proposal AIDA - WP2

- in WP2 'Common Software' will develop common tools for HEP community with focus on future accelerators, i.e. sLHC, ILC, CLIC (budget ~3FTEs*4 years)



Objectives

Task1: Coordination of Work package

- monitor the progress of the work in the work package
- coordinate and schedule the execution of the tasks and subtasks
- prepare progress reports – internal and on deliverables

Task2: Development of a geometry toolkit

- allow the description of complex geometrical shapes, materials an sensitive detectors
- provide interfaces to full simulation programs (Geant4), fast simulations, visualization tools and reconstruction algorithms
- allow for the misalignment of detector components
- provide an interface to calibration constants and conditions data

Task3: Development of generic reconstruction tools

- tracking toolkit based on best practice tracking and pattern recognition algorithms
- provide alignment tools
- allow for pile up of hadronic events
- calorimeter reconstruction toolkit for highly granular calorimeters based on Particle Flow algorithms

ILC computing on the Grid

- massive computing resources will only be available on the GRID
- -> already used by ILD and testbeams successfully
- VOs 'calice' and 'ilc' exist since 2005 in EGEE
- the ILC community is established in EGEE (2005-10) and will be present in EGI (2010-12)
- major Gris Sites do commit resources to ILC but no MoUs exist as in WLCG (yet)
- usually computing resources are not reserved but shared with other VOs - LHC in particular
- data storage needs commitments from sites - large testbeam data repositories have to allocated

CPU resources on the Grid

- ‘calice’:
  - grid sites: UK(8), FR(3), DE(2), JP(2), NL, CZ, ES
  - #Jobs(08/09): 149k + 62k + 33k = 244k (DE,FR,UK)
  - CPU time: 0.4 Mh + 0.2 Mh + 0.1 Mh = 0.7 Mh = 80y
- ‘ilc’:
  - grid sites: UK(18), FR(4), DE(4), ES(2), JP(2), IL(2), RO(1)
  - #Jobs(08/09): 534k + 399k + 194k = 1127 k (DE,UK,FR)
    - CPU: 1.8Mh + 1.8 Mh + 0.7 Mh = 4.3 Mh = 500y
  - O(1-2%) of total EGEE grid or O(10%) of a large LHC experiment used by ILC
  - need to make sites aware of ILC needs once LHC data taking has started

Data storage on the Grid

- example: Data stored at DESY Grid (on tape):
- ‘calice' – raw data ('Tier0') and MC
  - 55TB
- ‘ilc’:
  - MC (LOI mass prod. etc.) : 81TB
  - ‘ilc/eudet-jra1’: 4TB
  - ‘ilc/tpc’: 2TB

future needs have to be communicated ahead of time !

Summary & Outlook

- we have a common frameworks for the testbeams and the global ILD detector studies providing synergies for both communities
  - easy to switch between tasks as toolset is known
  - fast feedback of R&D results into global detector design
- benefited from EUDET project
- ILD/ILC software development is still a limited effort
  - identified ambitious task list for ILD software (TDR2012 !)
- -> what can be accomplished wrt. new common software tools for testbeams depends on the available effort/manpower (AIDA?)
- core software group is committed to further closely collaborate with testbeam community !