New developments in iLCSoft core tools

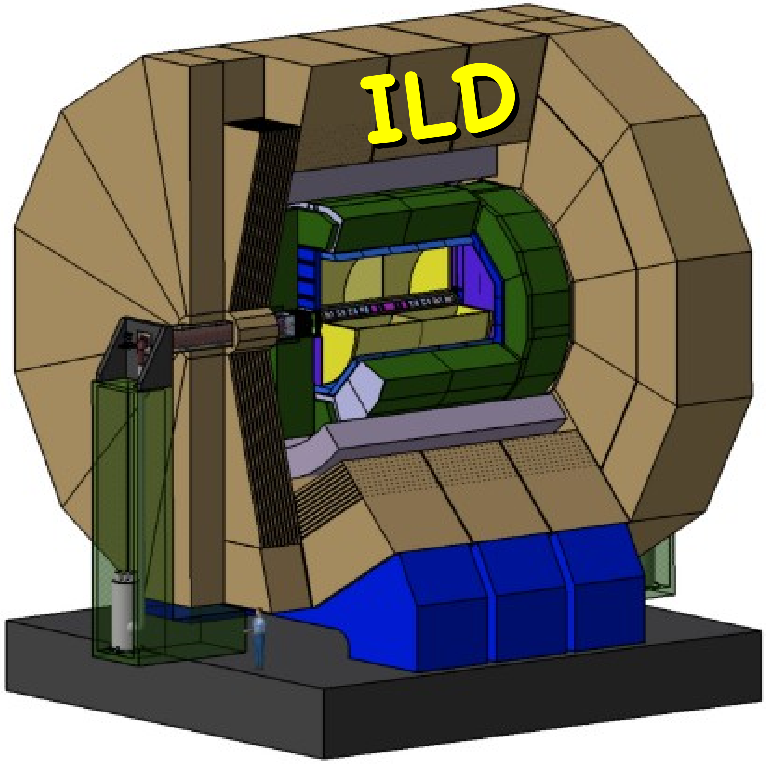
Frank Gaede, DESY

ILD Software and Integration Workshop

DESY, July 6-8, 2010

Outline

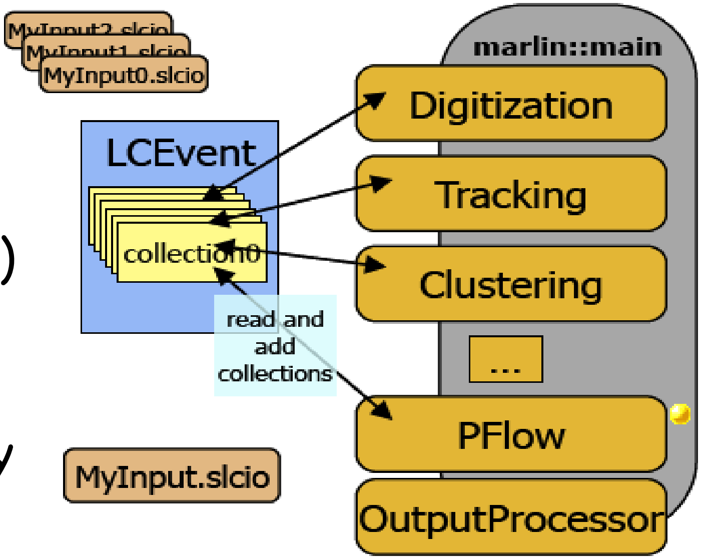
- Introduction
- new developments
  - ilcinstall
  - LCCD
  - LCIO
  - GEAR
  - Marlin
  - MarlinReco
  - Test system
- Summary & Outlook



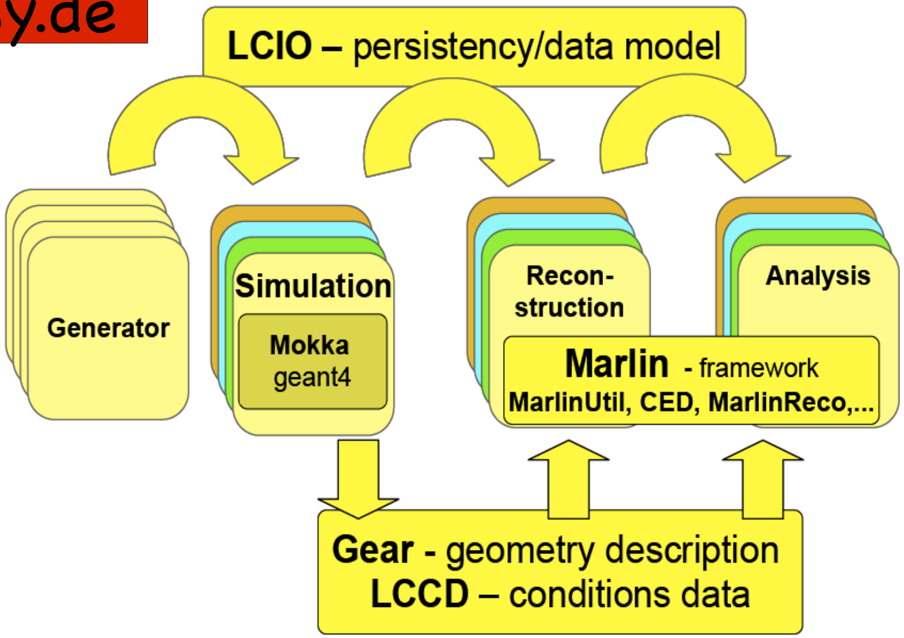
ILD Core Software Tools

http://ilcsoft.desy.de

- **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



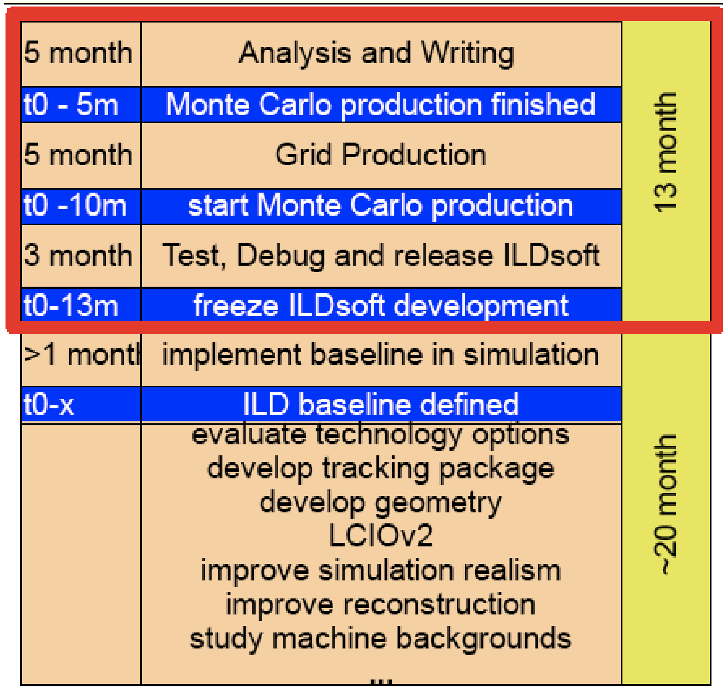
MyInput2.slcio
MyInput1.slcio
MyInput0.slcio
marlin::main
LCEvent
collection0
read and add collections
Digitization
Tracking
Clustering
...
PFlow
OutputProcessor
MyInput.slcio



LCIO – persistency/data model
Generator
Simulation
Mokka geant4
Recon-struction
Analysis
Marlin - framework
MarlinUtil, CED, MarlinReco,...
Gear - geometry description
LCCD – conditions data

- complete framework used in Monte Carlo & 'real experiments':
  - **ILD detector concept** studies
  - **Calice** calo testbeam
  - **LC-TPC** testbeam
  - EUDET - **Pixel Telescope**
- **synergies between testbeam and global detector optimization**

timeline for ILD core software development



5 month	Analysis and Writing	13 month
t0 - 5m	Monte Carlo production finished	
5 month	Grid Production	
t0 -10m	start Monte Carlo production	
3 month	Test, Debug and release ILDsoft	
t0-13m	freeze ILDsoft development	
>1 montl	implement baseline in simulation	~20 month
t0-x	ILD baseline defined	
	evaluate technology options develop tracking package develop geometry LCIOv2 improve simulation realism improve reconstruction study machine backgrounds ...	

- since LOI started to develop roadmap for ILD software
- need to prepare for DBD
  - Monte Carlo mass production
- time for core software development is in 2010 !

- merge goodies from JSF into framework
- develop a test system
- develop new GRID production system
- improve the geometry description
- improve the reconstruction (tracking & PFA)
- develop LCIOv2
- improve the simulation

Plans for ILDsoft:

- many points addressed
- see where we are in this workshop....

ILD software builds and installation

- **ilcinstall** tool: python scripts to download, build and install all ILD and external packages – incl. test beam
  - complete iLCSoft – provided geant4, root and mysql are installed
- used for
  - **iLCSoft releases** (ilcinstall release tag == ilcsoft release)
  - **reference installations** in afs
  - **grid installations** (all WLCG sites supporting VO ILC)
  - **binary tar-balls** (SL4/5)
- started to have more frequent 'developers' releases
  - goal: have defined and agree release schedule, so that groups can contribute their new developments on time
  - started to 'automize' software releases
  - not quite there yet ...

afs reference installations

- so far provided reference installations in afs (SL4/5) at
  - /afs/desy.de/group/it/ilcsoft
  - correct directory resolved through @sys mechanism from afs
    - eg: /afs/desy.de/group/it/.i386_linux26/ilcsoft
  - no longer fully supported - cannot cope with resolving 32bit/64bit, compiler version, CPU, kernel and OS version
- from current release v01-09 on use

/afs/desy.de/project/ilcsoft/sw/XXX/v01-09

XXX:	i386_gcc34_sl4	# i386 CPU, 32 bit, gcc3.4, SL4, SL5 and compatible
	x86_64_gcc41_sl5	# i686 CPU, 64 bit, gcc4.1, SL5 and compatible

- plan to have other OSs in the future (as needed)
  - note: old releases still at old path !

iLCSoft release v01-09

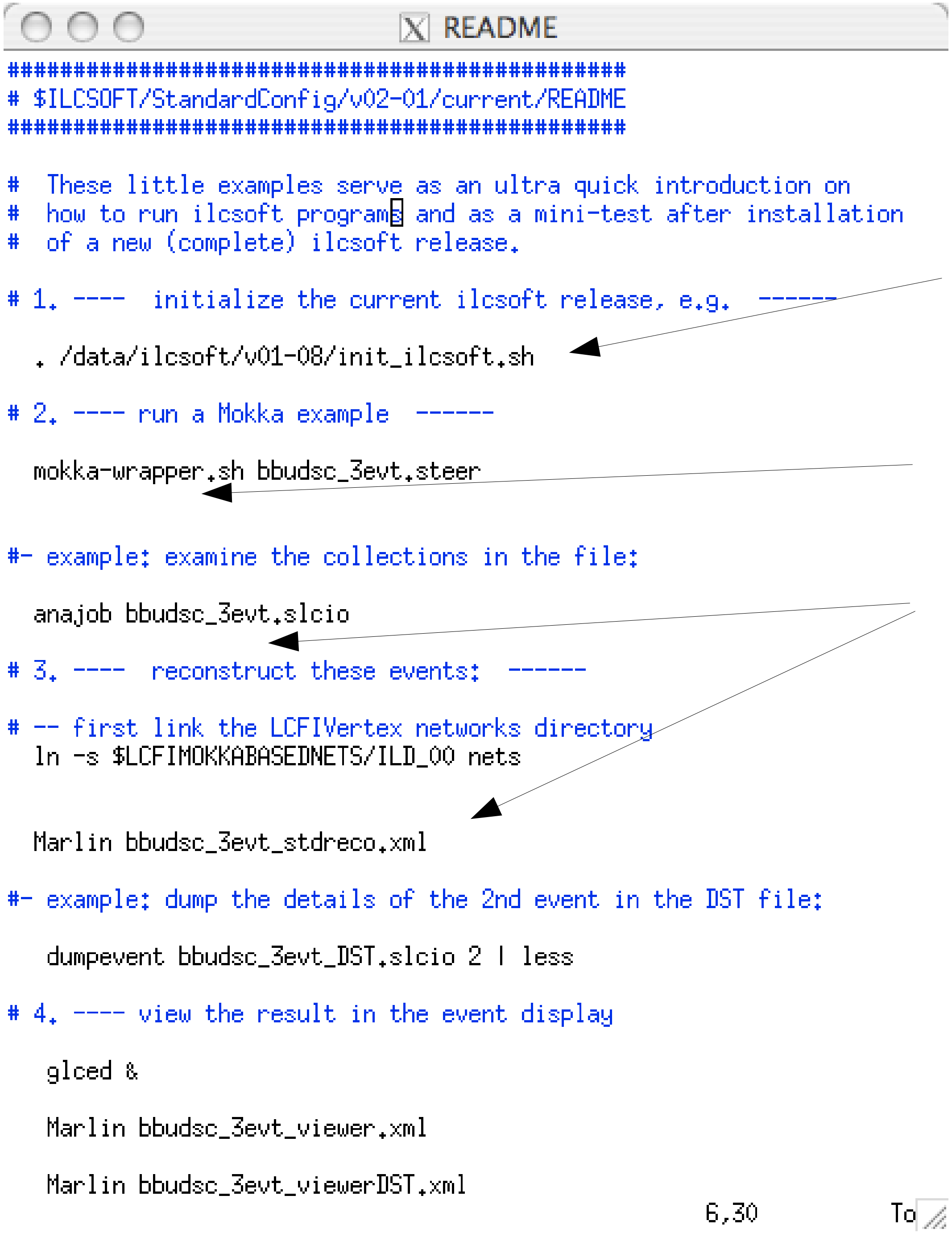
CED	v01-00 *
CEDViewer	v01-00 *
CLHEP	2.0.4.2
CMakeModules	v01-09
CondDBMySQL	ILC-0-8-1
Druid	**1.5 ***
Eutelescope	v00-02-02
LCFIVertex	v00-03-01
LCFI_MokkaBasedNets	v00-01
Marlin	v00-12
MarlinPandora	**v00-01 ***
MarlinReco	v00-18
MarlinTPC	v00-05-02
MarlinUtil	v01-00
Mokka	mokka-07-04
MokkaDBConfig	v02-01

Overlay	v00-07-03
PandoraPFA	v03-02-01
PandoraPFANew	**v00-02 ***
QT	4.2.2
RAIDA	v01-04-03
SiliconDigi	v00-04-02
StandardConfig	v02-01
cernlib	2006
dcap	1.9.5-5
gear	v00-14-01
gsl	1.8
java	1.6.0
lccd	v01-00
lcio	v01-51
mysql	5.0.45
root	5.26.00b

- many packages changed (wrt v01-08)
- **some new added**

*see dedicated talks

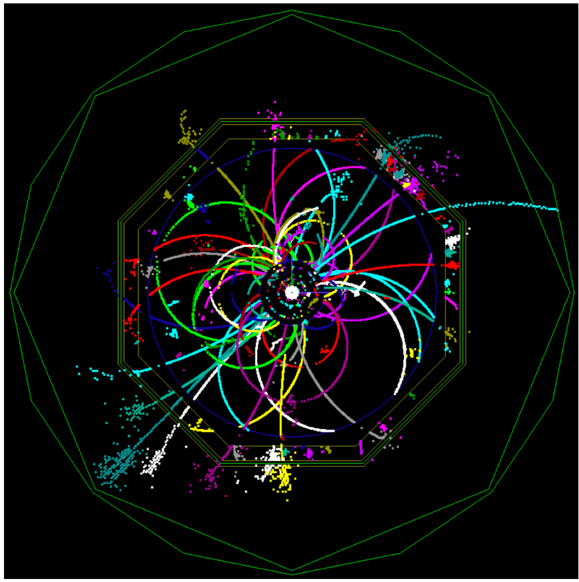
made running ilcsoft really easy



```
###############################################
# $ILCSOFT/StandardConfig/v02-01/current/README
###############################################

#  These little examples serve as an ultra quick introduction on
#  how to run ilcsoft programs and as a mini-test after installation
#  of a new (complete) ilcsoft release.

# 1. ----  initialize the current ilcsoft release, e.g.  ------

  . /data/ilcsoft/v01-08/init_ilcsoft.sh

# 2. ---- run a Mokka example  ------

  mokka-wrapper.sh bbudsc_3evt.steer


#- example: examine the collections in the file:

  anajob bbudsc_3evt.slcio

# 3. ----  reconstruct these events:  ------

# -- first link the LCFIVertex networks directory
  ln -s $LCFIMOKKABASEDNETS/ILD_00 nets


  Marlin bbudsc_3evt_stdreco.xml

#- example: dump the details of the 2nd event in the DST file:

   dumpevent bbudsc_3evt_DST.slcio 2 | less

# 4. ---- view the result in the event display

   glced &

   Marlin bbudsc_3evt_viewer.xml

   Marlin bbudsc_3evt_viewerDST.xml
```

[allready in v01-08 !]

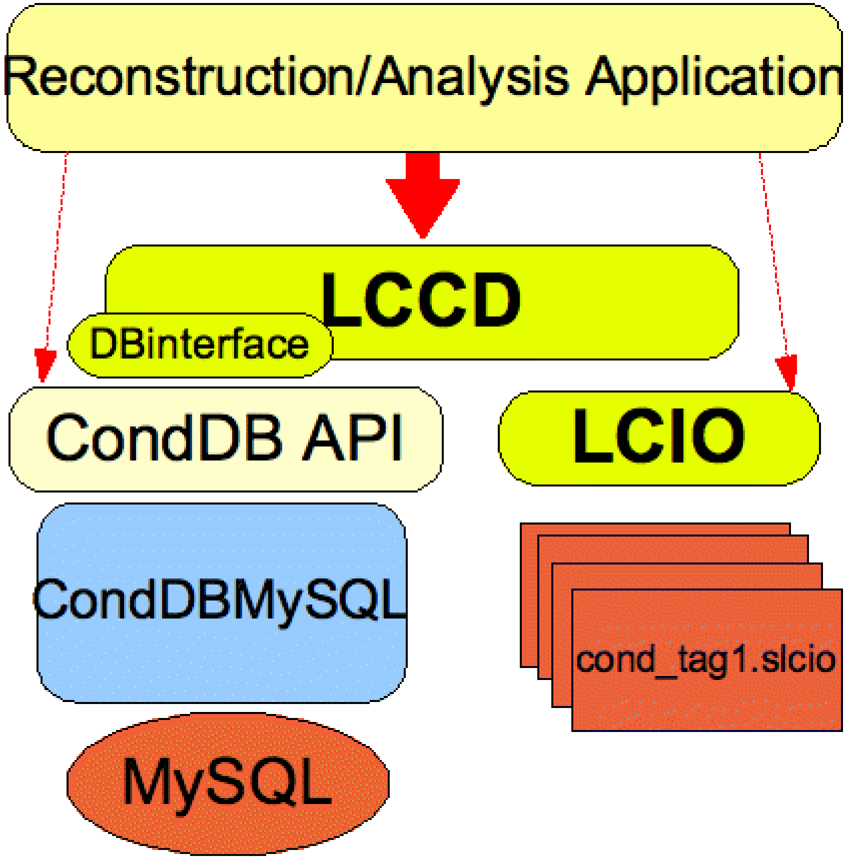
- new initialization script autogenerated from ilcinstall
- new script mokka-wrapper.sh to run Mokka w/ local database (db-dump in release)
- all binaries and libs available



LCCD v01-00

S.Aplin

- conditions data toolkit
  - abstract interface to cond. data
  - MySQL-DB, LCIO files,...
- used by Calice and LC-TPC
- improved robustness for missing conditions data
  - user now can register a default collection that is returned if nothing found (i.e not in DB)
  - originally not foreseen
  - request from the 'real world'



•LCCD v01-00 needs version CondDBMySQL_ILC-0-8 or higher !

•CondDBMySQL maintained by Calice

towards LCIOv2

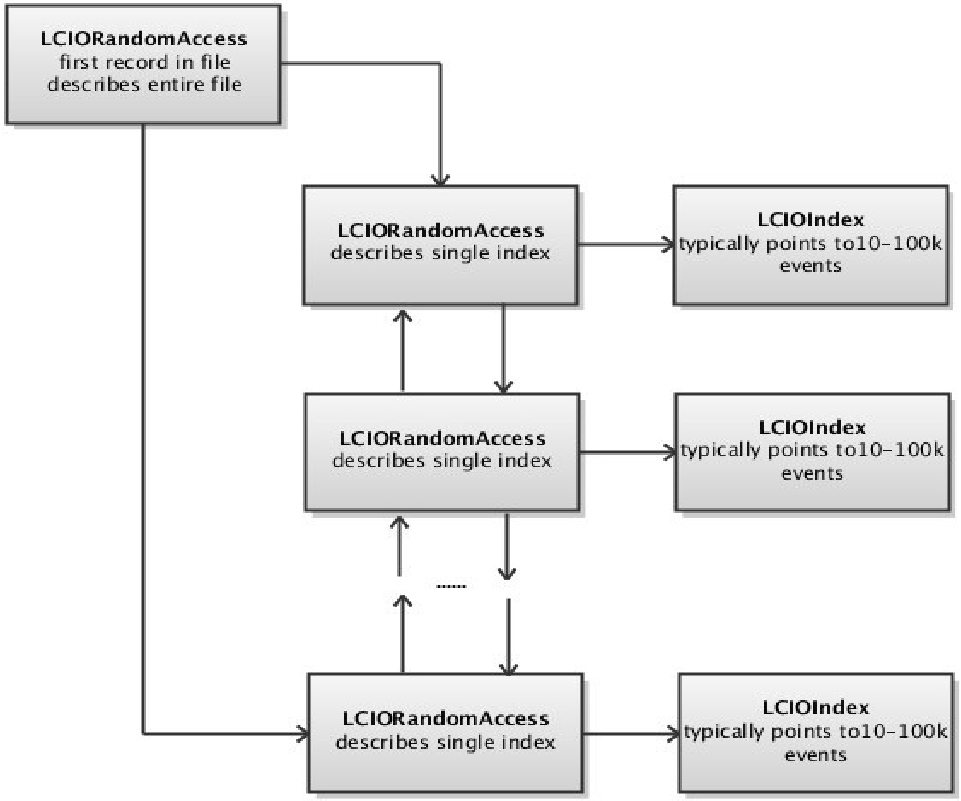
- LCIO provides a rather complete event data model and has been used successfully in SID and ILD LOI mass production and in various R&D testbeam programs
- LCIOv2 needs to be backward compatible and should provide some new features
  - **direct access to events**
  - **partial reading of events**
  - **splitting of events over files**
  - **(storing of arbitrary user classes)**
  - **simplify using LCIO with ROOT**
    - (ROOT macros, TTreeViewer, I/O (?) ,...)
  - **improving the event data model**
    - (1d,2d hits, tracks/trajectories)

LCIO release v01-51

- LCIO v01-51 released
  - direct access
  - improved EDM
  - new ostream operators<<(...) in C++
    - cout << ((MCParticle*) c->getElementAt(i)) << endl ;
  - bug fixes and improvements for developers
- 'first' step towards LCIOv2
- will have LCIO developers meeting after the workshop
  - work on EDM and further improvements

direct access to LCIO events

- direct access to LCIO events needed:
  - overlay of random background events
  - physics analysis - reading of pre-selection
  - so far available through fast skip or creation of TOC on opening (slow)
  - → introduced two additional records **LCIORandomAccess/LCIOIndex**

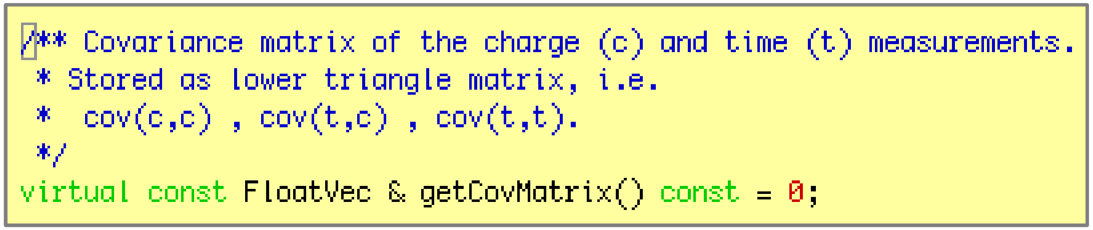


  - records written at end of file on close()
  - can append to file
  - **can add direct access to existing file**
    - if opened in append on writable file system (not tape)
- released in v01-51

improved Tracker Hit classes

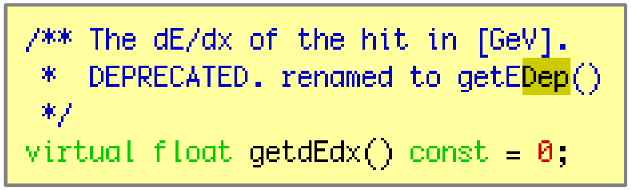
released in v01-51

- TrackerPulse
  - added covariance (error) matrix for charge and time measurements

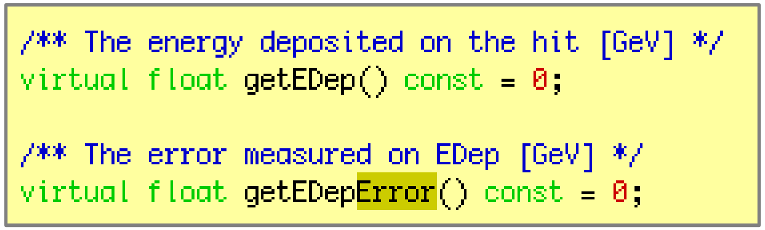


```
/** Covariance matrix of the charge (c) and time (t) measurements.
 * Stored as lower triangle matrix, i.e.
 *  cov(c,c) , cov(t,c) , cov(t,t).
 */
virtual const FloatVec & getCovMatrix() const = 0;
```

- (Sim)TrackerHit
  - renamed dEdx to EDep - deposited energy
    - dEdx methods are deprecated: they still can be used but result in a printed warning ...
  - added EDep to TrackerHit
    - measurement error



```
/** The dE/dx of the hit in [GeV].
 *  DEPRECATED. renamed to getEDep()
 */
virtual float getdEdx() const = 0;
```



```
/** The energy deposited on the hit [GeV] */
virtual float getEDep() const = 0;

/** The error measured on EDep [GeV] */
virtual float getEDepError() const = 0;
```

a ROOT dictionary for LCIO

- LCIO now comes with a ROOT dictionary for all LCIO classes (optional) - with this one can:

  (allready in v01-12-01)

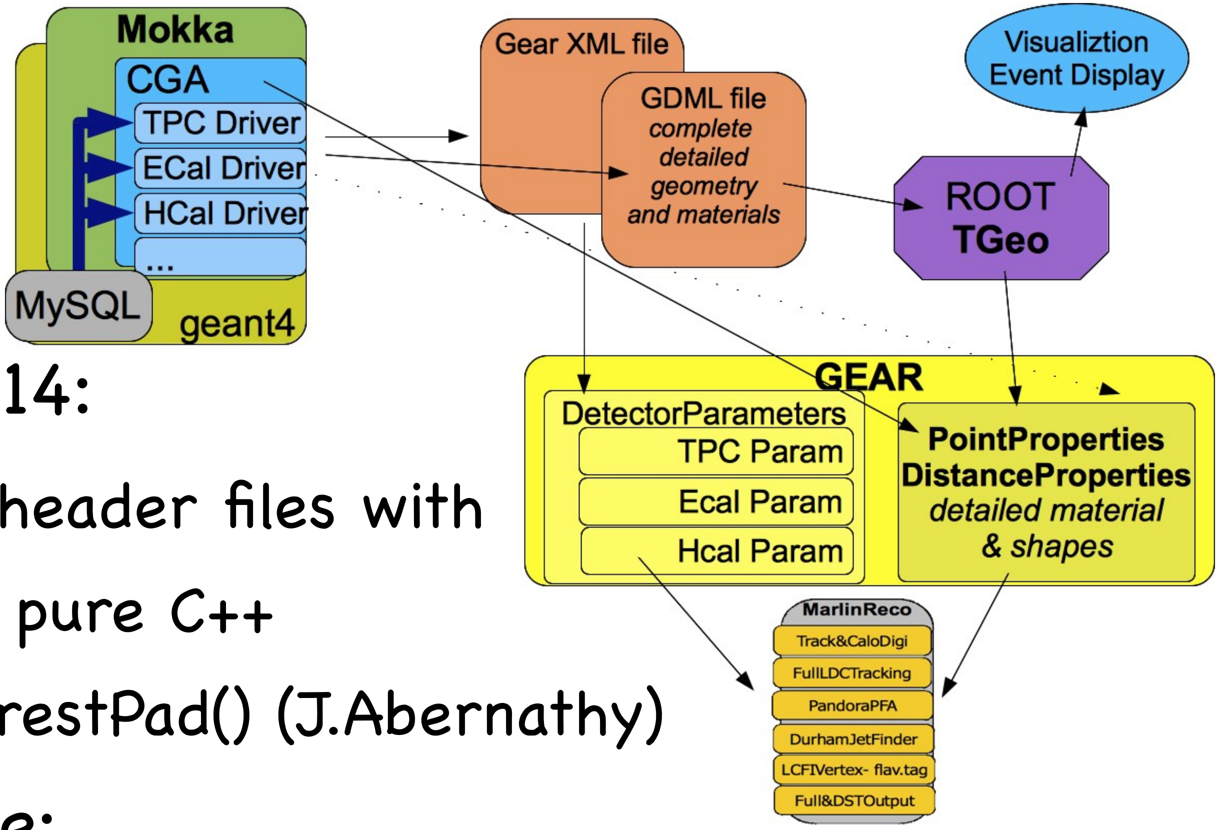
  - use LCIO classes in ROOT macros
  - write simple ROOT trees, e.g. std::vector<MCParticleImpl*>
  - use TTreeDraw for quick interactive analysis of LCObjects:

```
//---gamma conversions:
TCut isPhoton("MCParticlesSkimmed.getPDG()==22" ) ;
LCIO->Draw("MCParticlesSkimmed._endpoint[][0]:
                MCParticlesSkimmed._endpoint[][1]",isPhoton ) ;
```

  - write complete LCIO events in one ROOT branch
  - see: $LCIO/examples/cpp/rootDict/README for details & help

- -> we are interested in feedback from the users if this provides already the requested features

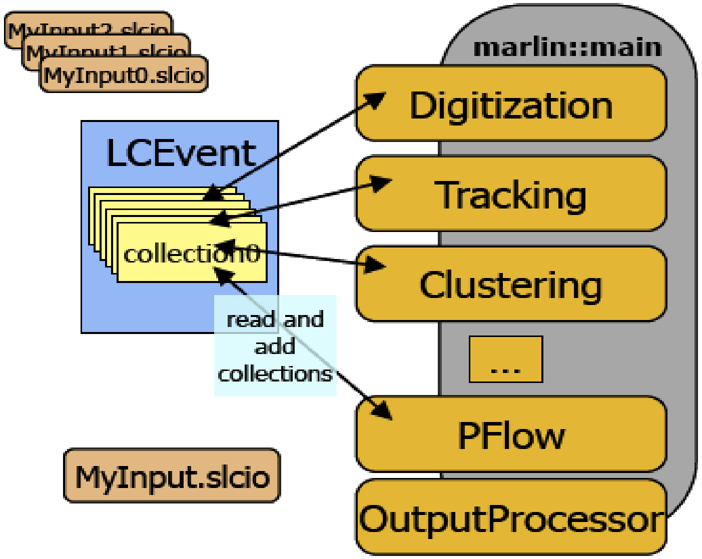
GEAR



- new features in v00-14:
  - removed generation of header files with aid (Java tool) -> made pure C++
  - improved TPCP.::getNearestPad() (J.Abernathy)
- not yet in this release:
  - Point/DistanceProperties implemented with TGeo (see talk by A. Muennich)
    - exists in branch - to be released soon
  - Mokk-CGA implementation released in Mokka (see talk P .Mora de Freitas)

Marlin

- new features in v00-12:
  - processors can have local Verbosity parameter
    - can turn of messages from other processors for debugging
  - exit if processor specified in steering file not found
  - processor return values have to be set if used in logical expression in steering file
    - patch by Phillipe Klenze
  - minor bug fixes (gcc4.x)



to do for next release: introduce command line parameters – **from JSF framework**

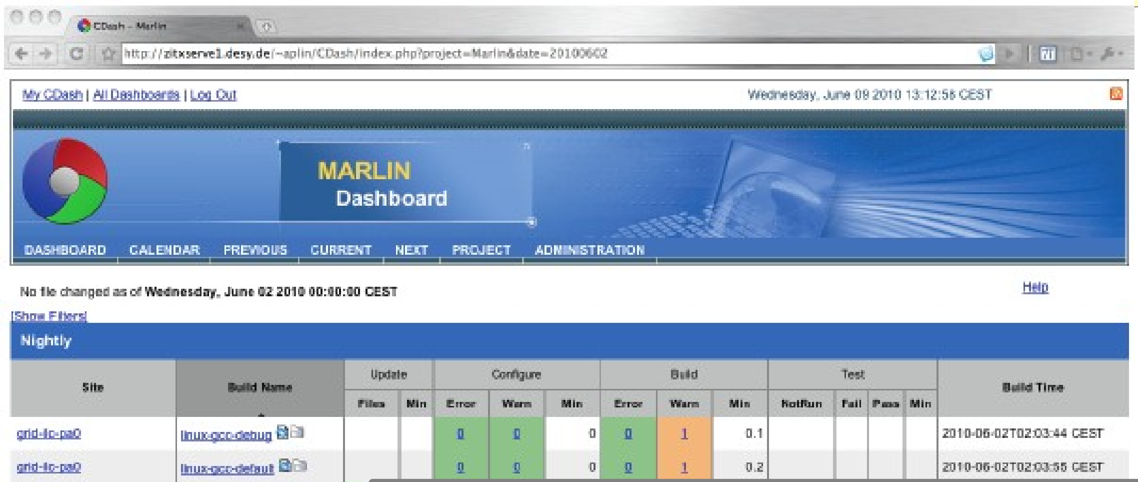
MarlinReco

- added new package Tracking/KinkFinder (M.Thomson, J.Marshall)
- 'new' package BCalTagEfficiency (J.List,M.Berggren)
  - should be run stdreco in the future (no bcal in pandora)
  - moved from MarlinAna to MarlinReco
- improved compatibility for gcc 4.x
  - some bug fixes:
    - SimpleMuonDigi: added missing output relation collection to the event
    - SimpleMuonDigi: decoding of calorimeter layout in CalorimeterHit::type
    - fixed bug in TPCDigitizer with hit-MCParticle association
    - PFOID fixed parameter description in documentation
    - TPCDigiProcessor fixed a bug in the logic when checking for hits from the same MCParticle

Test system for iLCSoft

- started to develop test system (S.Aplin) - goal to have
- 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,...
- use CTest (CMake) and CDash (webinterface)
  - started to add unit-tests to some packages
  - integration tests (build/install) run in nightly builds
  - physics tests: DESY summer student program 2010

cdash: webinterface to test system



CDash - Marlin

http://zitxserve1.desy.de/~aplin/CDash/index.php?project=Marlin&date=20100602

My CDash | All Dashboards | Log Out

Wednesday, June 09 2010 13:12:56 CEST

MARLIN Dashboard

DASHBOARD CALENDAR PREVIOUS CURRENT NEXT PROJECT ADMINISTRATION

No file changed as of **Wednesday, June 02 2010 00:00:00 CEST**

[Show Filters]

Help

Nightly

Site	Build Name	Update		Configure			Build			Test				Build Time
		Files	Min	Error	Warn	Min	Error	Warn	Min	NotRun	Fail	Pass	Min	
grid-ilc-pa0	linux-gcc-debug			0	0	0	0	1	0.1					2010-06-02T02:03:44 CEST
grid-ilc-pa0	linux-gcc-default			0	0	0	0	1	0.2					2010-06-02T02:03:55 CEST
grid-ilc-pa0	linux-gcc-default-tests													
Totals	3 Builds													

No Continuous Builds

No Experimental Builds

No Coverage

No Dynamic Analysis

Kitware

Generated in 0.073 seconds

- used for ilcSoft
- to be used for Calice
- other packages can be added

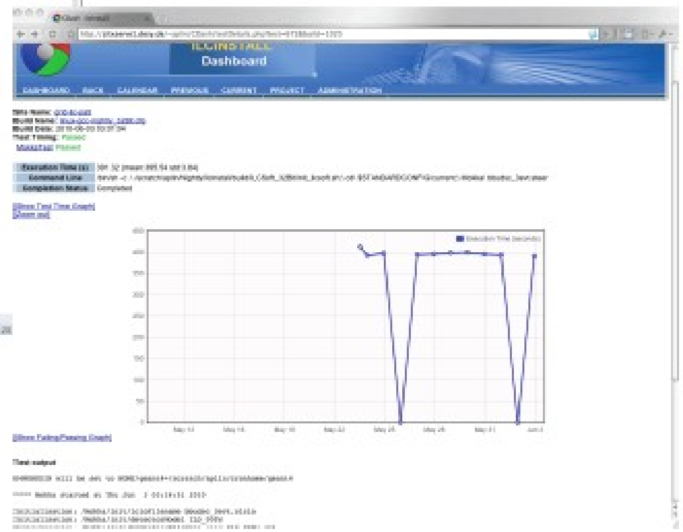
ILCINSTALL Dashboard

5 tests passed.

Name	Status	Time Status	Time
[illegible]	Passed	Passed	[illegible]
[illegible]	Passed	Passed	[illegible]
[illegible]	Passed	Passed	[illegible]
MarlinTest	Passed	Passed	[illegible]
MokkaTest	Passed	Passed	[illegible]

Kitware

Automatic Email notification if and when build or tests fail.
Developers may subscribe to several projects to keep an overview



Currently the test available concentrate on package specific tests, and integration tests
i.e. Mokka – MarlinReco

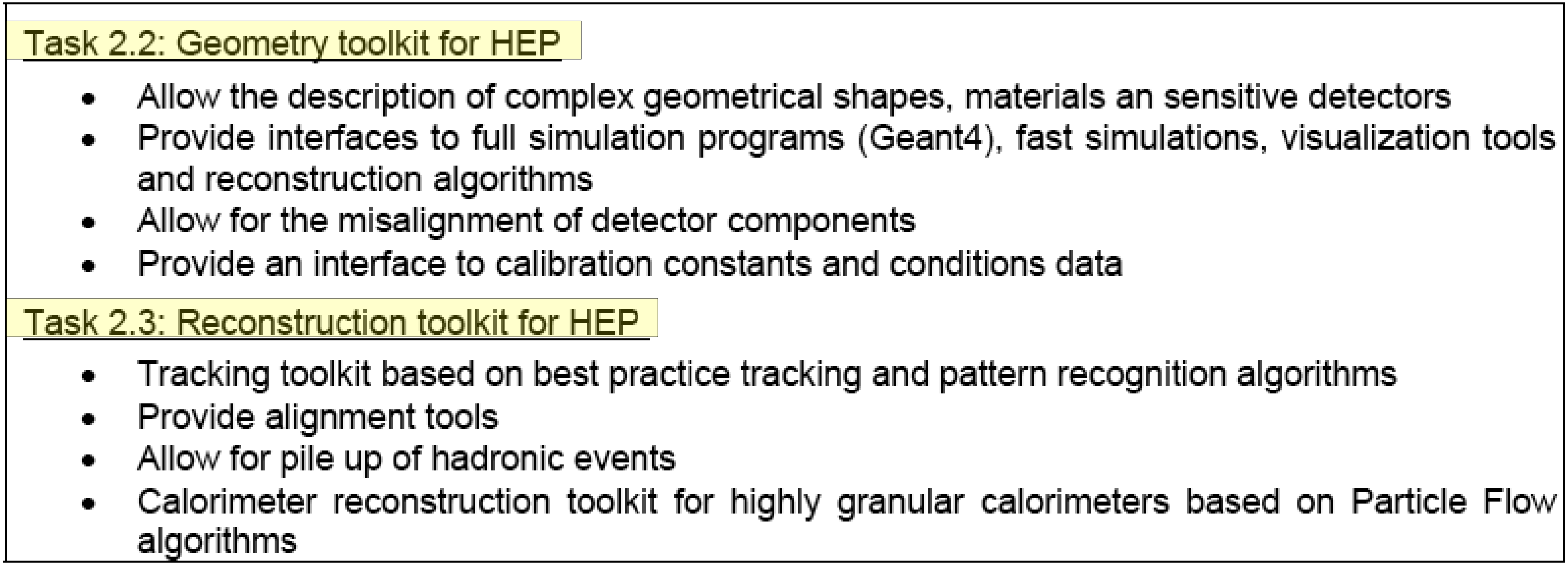
Plans to implement physics based tests to monitor performance

The AIDA project

http://www.cern.ch/aida

Advanced European **I**nfrastructures for **D**etectors at **A**ccelerators

- EU project that addresses infrastructures required for the development of detectors for future particle physics experiments:
  - sLHC, ILC/CLIC, neutrino facilities, B-factories
- project duration: 4 years - starts: Feb. 2011
- WP2: common software development for HEP



Task 2.2: Geometry toolkit for HEP

- 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

Task 2.3: Reconstruction toolkit for HEP

- 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

Summary & Outlook

- since LOI we started to work towards a roadmap for ILD software development wrt. DBD
- major developments in core software tools should be done mainly in 2010
- some significant improvements in core tools with latest release v01-09
- still quite a few things to do
  - LCIOv2, GEAR, test system, ...
- however: ILD software is fully functional for any ongoing studies !
- need to also start seriously looking into development of new tracking software

additional material

ROOT I/O for LCIO

- user request to have closer link of LCIO to ROOT
  - use LCIO classes in ROOT macros (former GLD groups)
  - have fast interactive analysis with ROOT tree
- investigate the optional use of ROOT I/O for LCIO
  - would provide 'missing features': direct access, partial reading and splitting of events (and streaming of user classes)
- created experimental branch in cvs (rio_v00-00)
  - create ROOT dictionary w/ help from ROOT team (A.Naumann)
  - implemented index based pointers for C++
  - needed some changes to LCIO classes: LCTCollection<T>, std::vector as members, ,...
  - can create almost complete copies of LCIO DST in ROOT
    - no subcollections (pointers only) yet
  - streaming mode for Marlin under development
- see: talks at ILD software working group meetings for details
- still some issues to resolve (interface to Java !!)

storing of arbitrary user classes

- LCIO event data model rather complete – but also clear need for storing user defined information
  - LCGenericObjects can store almost arbitrary data structures based on ints, floats and doubles
    - files can be read w/o any additional code (dictionary)
    - small performance penalty
    - extensively used in LCCD (conditions data) by testbeams
- occasional user request for 'natively' storing arbitrary user classes in LCIO
  - possible in principle with LCIO/SIO (not documented and somewhat 'discouraged') – would come 'for free' w/ ROOT I/O
- IMHO: success of LCIO is to a large extend due to the slightly restrictive definition of the event data model i.e. the interfaces between modules/processors

partial reading & splitting of events

- needed for performance and cost (disk space) issues:
  - read only objects of interest in analysis (PandoraPFOs)
  - store simulation and reconstruction output in separate files
- main obstacle: need pointer/reference mechanism across I/O records and files
  - not available in SIO now and can't use TRefs in ROOT
- need index based pointers independent of I/O, e.g.:
  - long64 index = HASH(collName) << 32 | collIndex
- experimental C++ version exists in ROOT I/O branch for partial reading of events (not yet file splitting)
  - need further testing & implementation in SIO (also Java)
  - need extension of LCIO::Reader interface

Improving the LCIO event data model

- suggested improvements to the event data model:
- 1D, 2D tracker hits
  - LCIO (Sim)TrackerHit is a 3D space point - whereas actual measurements are either 1D (strip) or 2D (TPC) where the detector surface (line) provides the additional geometry information
- Track
  - currently Track has pointers to all TrackerHits and one set of (Helix) parameters
  - generally one wants to have multiple fits for one set of hits, e.g. at the IP or at the face of the calorimeter
  - Trajectory could be introduced as high level convenient view to these fits
  - currently not straight forward (though possible) to store kinks in LCIO

- details are coupled to development in tracking code
- hope to make progress at this meeting (LCIO meeting friday)
- also user feedback welcome

Digitization & Reconstruction in Marlin

- VXD, SIT, FTD, SET, ETD
  - smearing of 3D space points according to detector resolutions as established by R&D groups
- TPC hits
  - smearing of 3D space points - taking into account drift distance, polar and azimuthal angle of track
  - parameterization from TPC R&D groups
- ECal, HCal, LCal, Bcal, LHCal, Muon Calo hits
  - calibration (single particle resolution)

- Tracking*
  - standalone tracking in Silicon detectors and TPC - **MarlinReco-FullLDCTracking**
  - Kalman filtering: wrapped f77 code from LEP
- Particle Flow Algorithm*
  - **PandoraPFA**: best PFA to date
- JetFinder
  - Durham jet finder (run for 2-6 jets)
- Flavour Tagging*
  - **LCFIVertex** package: ZVTop, ZVRes + Neural Network Fl.Tag
- DST Maker
  - ReconstructedParticles, Jets, Tracks and Clusters (25k/evt)

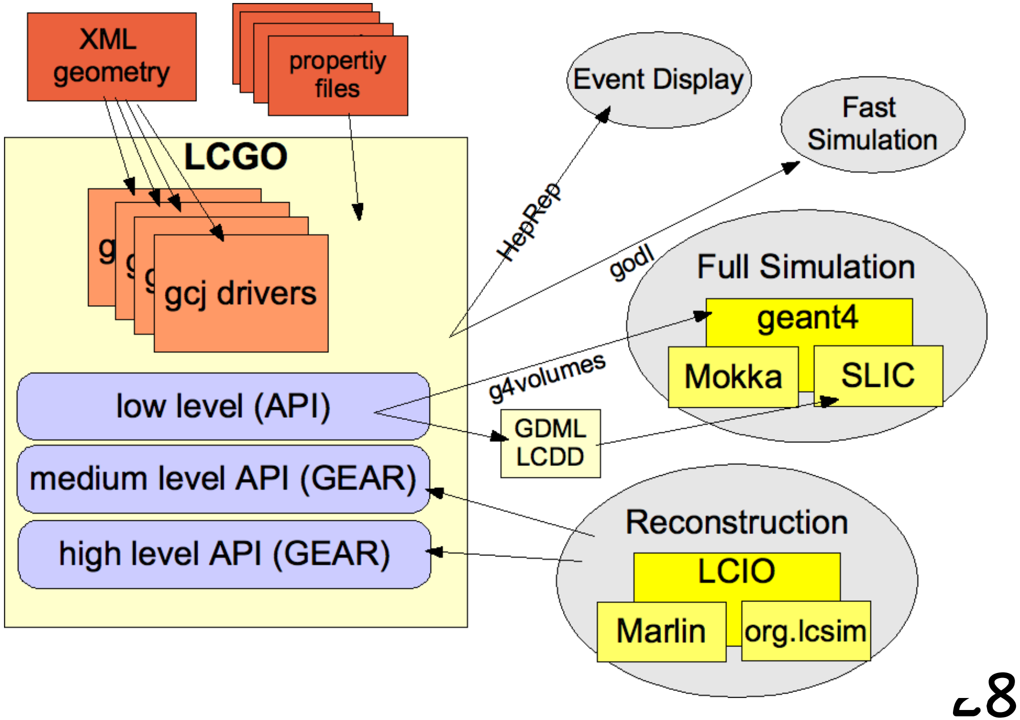
* see dedicated talks this workshop

goal: new generic geometry Toolkit

- current geometry system could be improved:
  - no user parameters
  - one packet that feeds into
  - full simulation, i.e. geant4
  - 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

Development of such a toolkit would be part of AIDA fp7 project

- features needed:
  - allow for misalignment
  - small memory footprint
  - local to global (cellID-position)
  - fast navigation (?)
  - access to detailed material
- could base on ROOT-TGeo...



example: LCGO conceptual