ILD Core Software

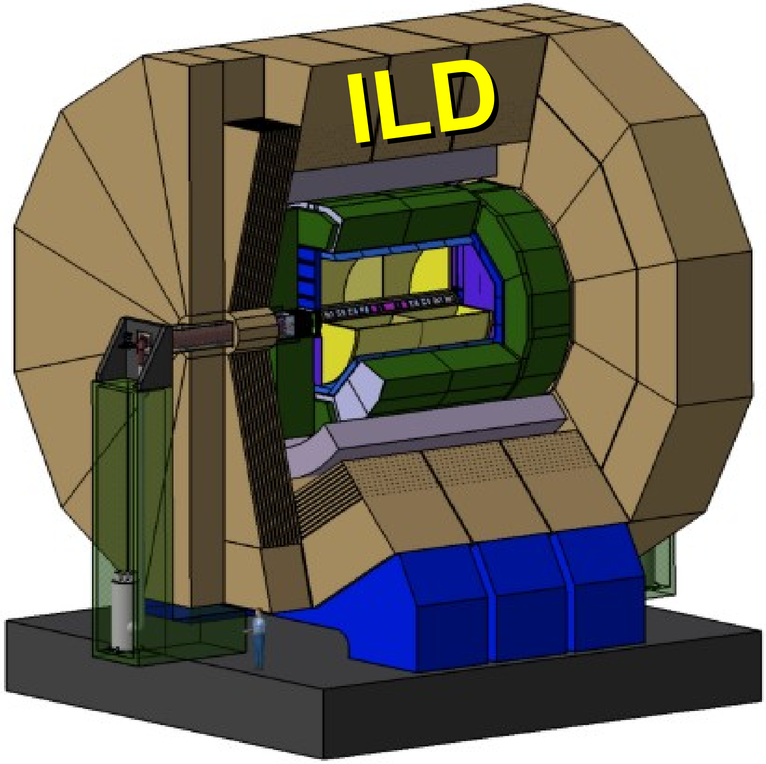
Overview, Status and Plans

Frank Gaede
DESY

International Linear Collider
Workshop 2010
Beijing, March 26-30, , 2010

Outline

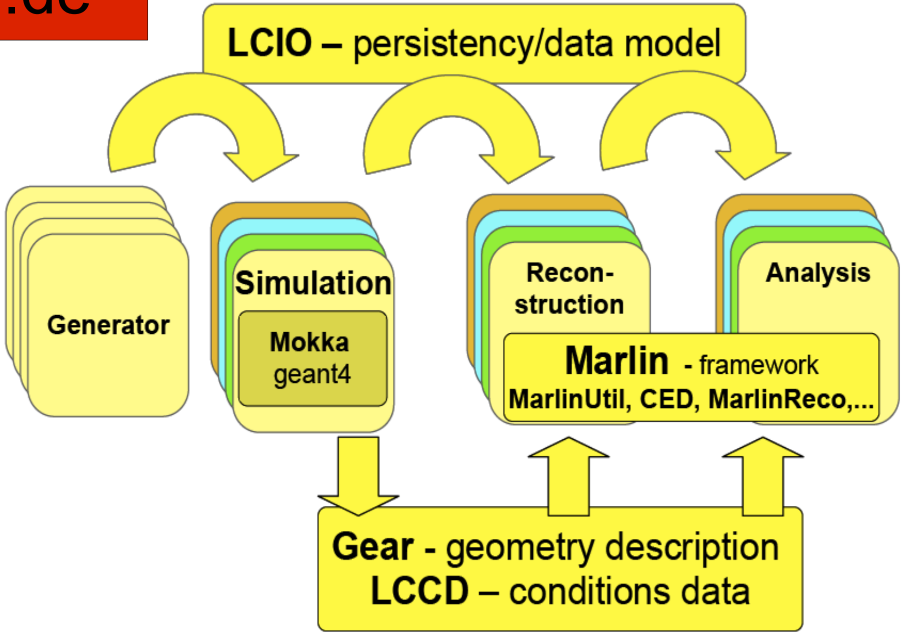
- Introduction & Overview
  - core – simulation - reconstruction
- new developments
  - installing & runnning ilcsoft
  - Grid tools
  - LCIOv2
  - geometry
- Summary



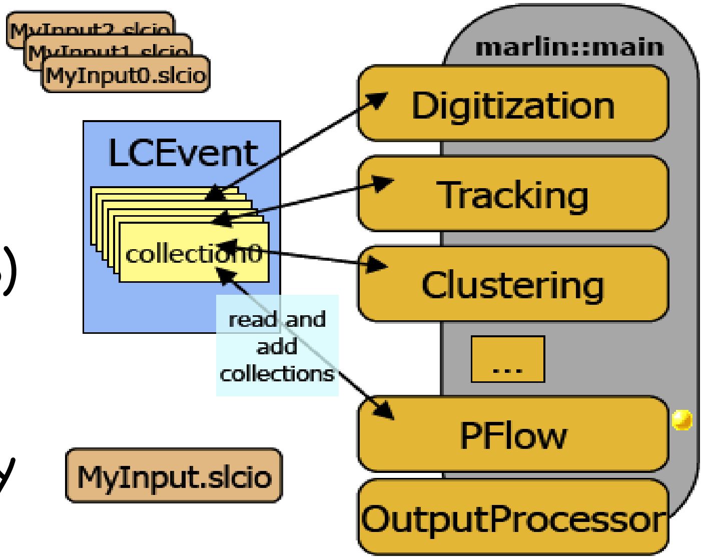
ILD Core Software Tools

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



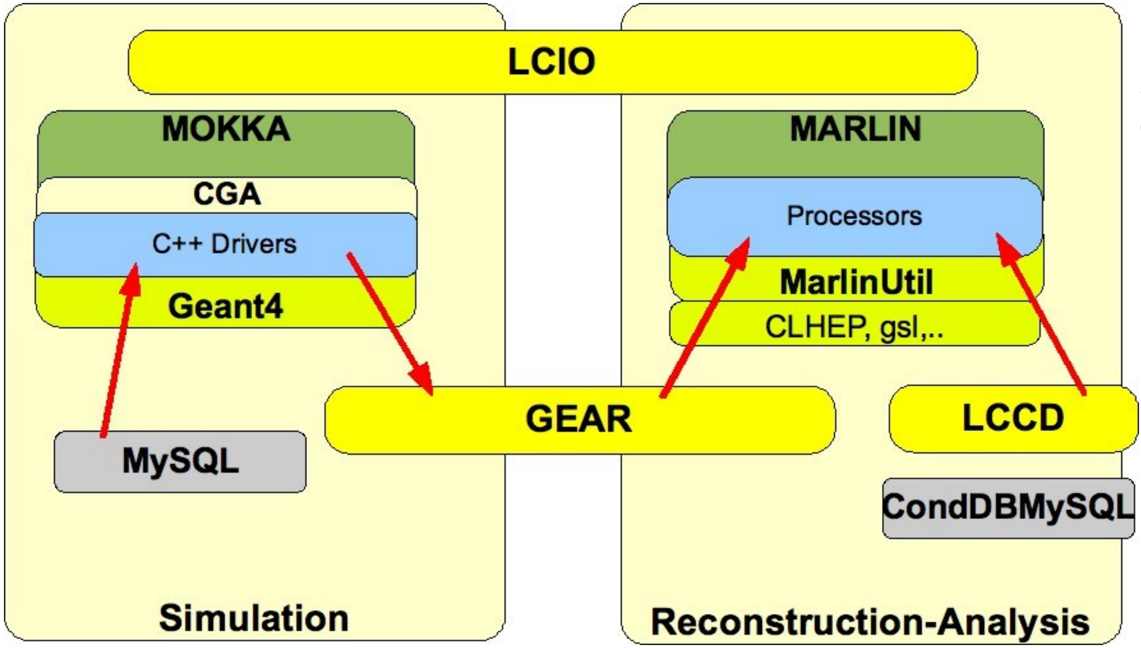
- complete framework used in Monte Carlo & 'real experiments':
  - **ILD detector concept** studies
  - **Calice** calo testbeam
  - **LC-TPC** testbeam
  - EUDET – **Pixel Telescope**
- **synergies between testbeam and global detector optimization**

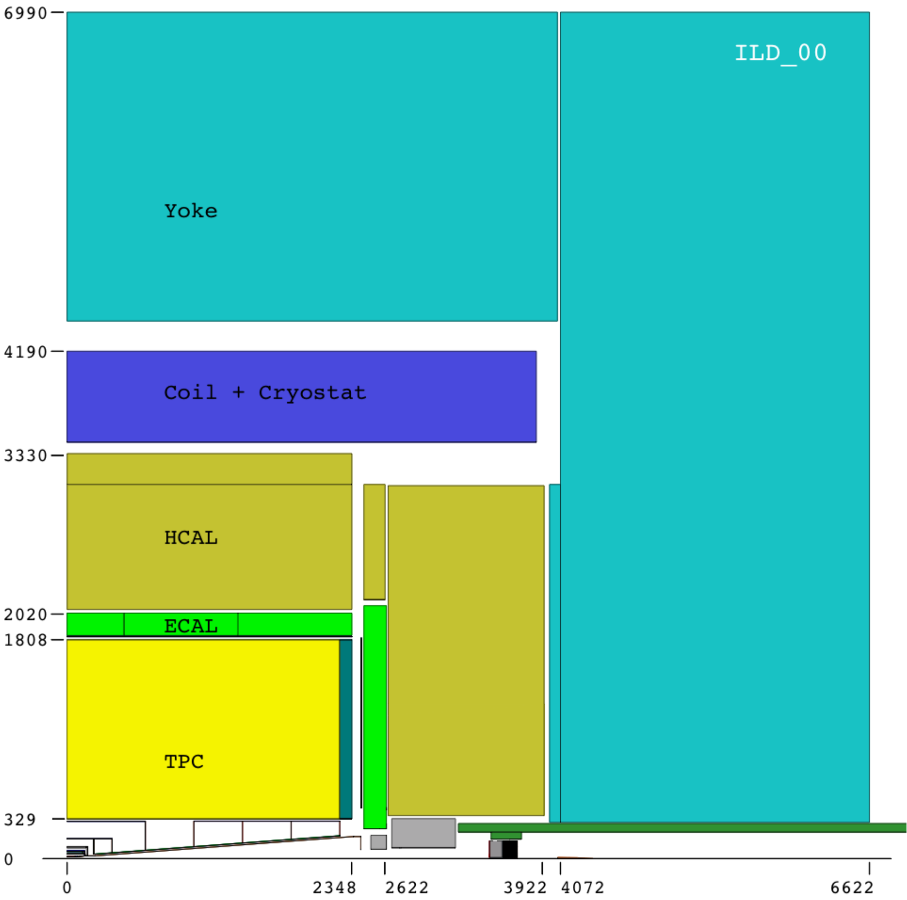


Mokka Simulation ILD

Frank Gaede, DESY, LCWS 2010, Beijing 26-30.03.2010

- defined 'ILD simulation reference model' for LOI mass production
- engineering level of detail for most subdetectors:
  - support structures
  - cracks
- further improve realism for **DBD**
  - -> see talk A.Miyamoto





Mokka writes out GEAR xml files with complete geometry and material parameters that are need for reconstruction and analysis

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

ILD software builds and installation

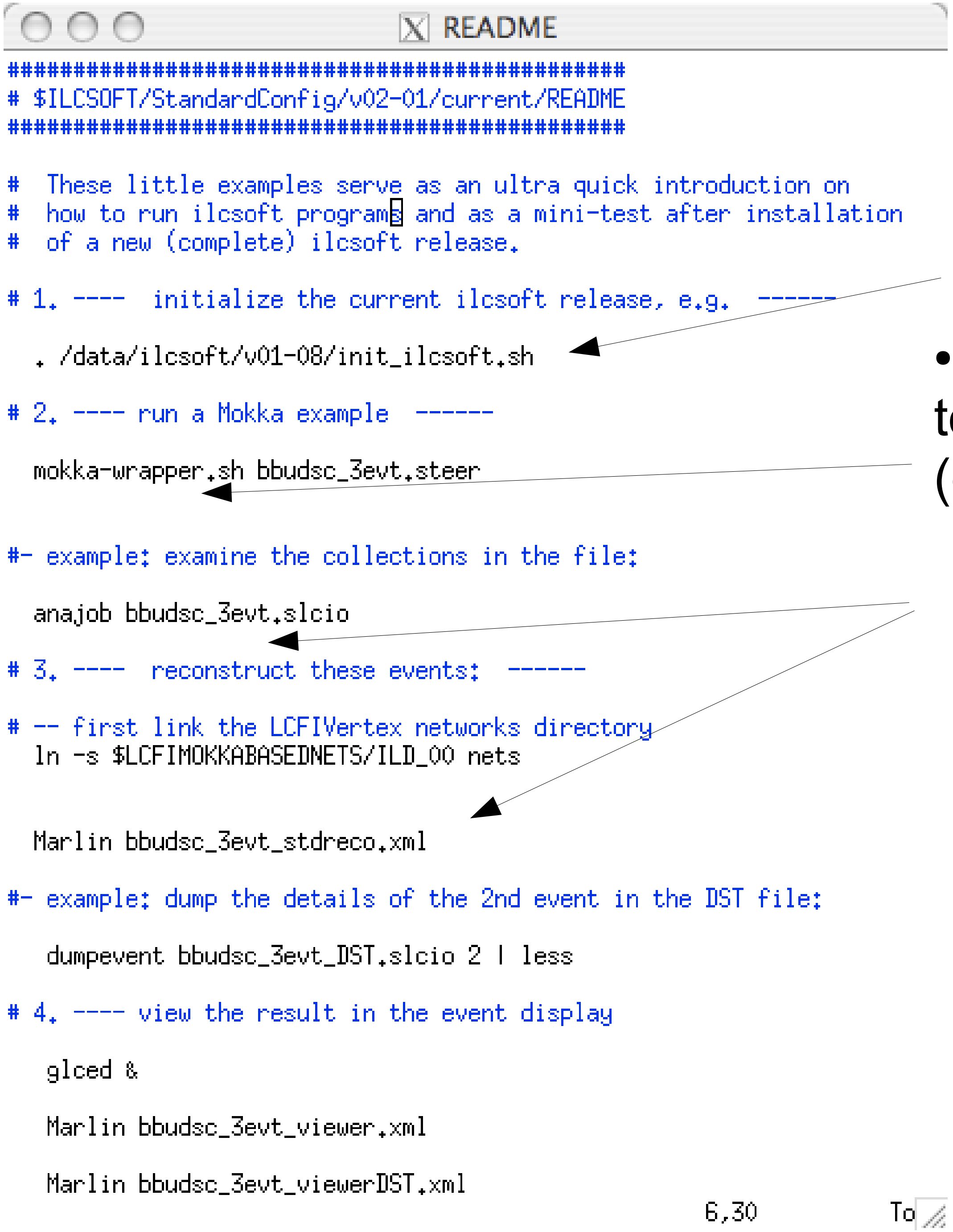
- **ilcinstall** tool: python scripts to download, build and install all ILD and external packages - incl. test beam
  - 'edit and start configure script - go to lunch - run ILD software'
  - on 'scratch' disk - provided geant4, root and mysql are installed
- used for
  - **reference installations** in afs (SL4/5)
  - **grid installations** (all WLCG sites supporting VO ILC)
  - **binary tar-balls** (SL4/5)
- started to have more frequent 'developers' releases
  - goal: have defined and agree release schedule, so that groups can contribute their new developments on time
  - need to 'automize' software releases...

current release: ilcsoft v01-08-01

/afs/desy.de/group/it/ilcsoft/v01-08-01

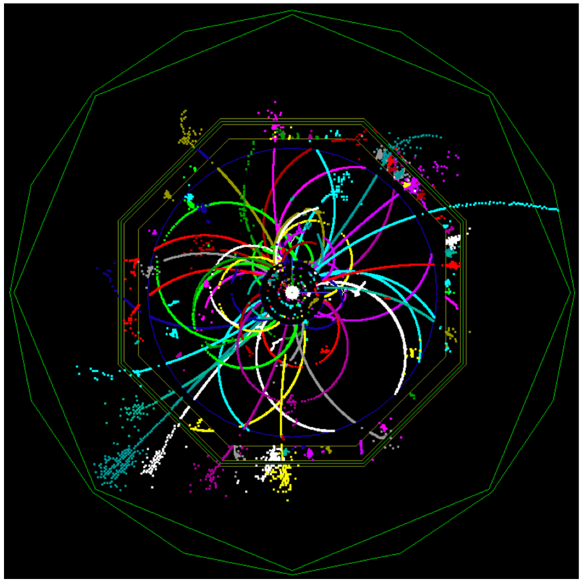
- works on 64-bit:
  - 32 bit compatibility mode and natively
    - provided you have a cernlib 64-bit binary
- made installation and running easier
- root dictionary in LCIO
- tracking code improved since LOI (bg studies)
- new Mokka detectors (forward, dHCal,...)
- bug fixes in MarlinReco, LCIO, ...

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
```

- 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

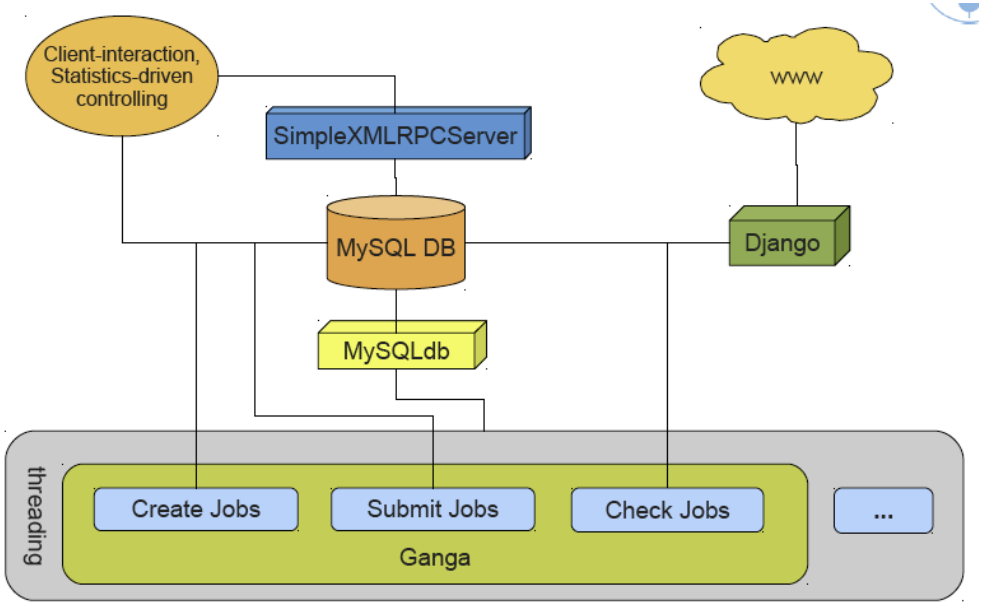


Grid software installations

- virtual organization ILC now shared by ILD, SID and CLIC (in Europe and Asia)
- need to organize software installations :
  - use common software where possible: LCIO, ROOT, geant4,...
    - communicate about installed (and de-installed versions)
    - use ilc-vo-support@desy.de for communication
  - software installations in separate directories:
    - $VO_ILC_SW_DIR/[ilcsoft, sidsoft, lcdsoft]/lcio/v01-XX
- VO ILC in US exists with independent members
  - should try and come to agreement on common membership
  - or optionally have only one common VO
  - ... how to achieve this ?

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
- submission and monitoring of Grid jobs
- data catalogue based in database



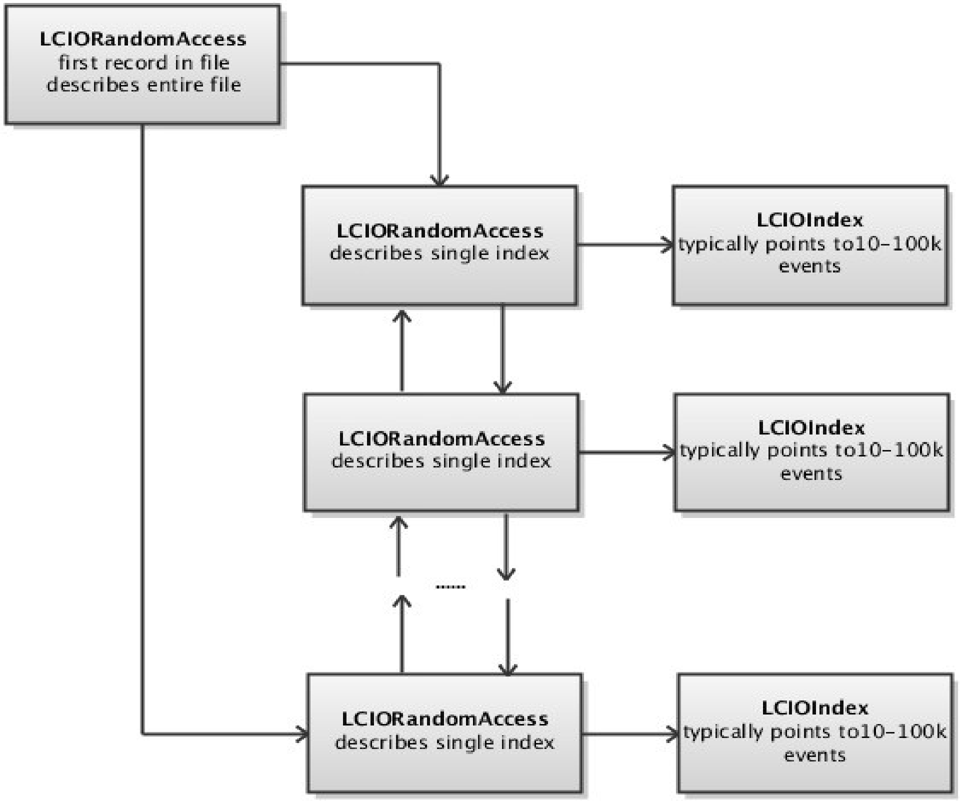
see talk by S.Aplin

also collaborate with CLIC group on DIRAC (see talk A.Sailer)

towards LCIO v2

- 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
- user defined classes stored in LCGenericObject
- runtime extensions (C++): attach arbitrary C++ types to any LCObject and N-to-M relationships (not used frequently !?)
- current I/O: SIO compression one event per record
- planned and ongoing improvements:
  - **direct access to events (now only via fast skip ot TOC creation)**
  - **partial reading of events (e.g. only PandoraPFOs)**
  - **splitting of events over files (sim, rec, DST w/o dublication)**
  - **storing of (arbitrary) user classes**
  - **use LCIO with ROOT (ROOT macros, TTreeViewer, I/O (?), ...)**
  - **improving the event data model (1d,2d hits, tracks/trajectories)**

direct access to LCIO events



- direct access to LCIO events needed:
  - overlay of random background events
  - physics analysis - reading of pre-selection
  - debugging
- now available through fast skip or creation of TOC on opening (slow)
- proposed extension of LCIO/SIO (T. Johnson):
  - add two additional records LCIORandomAccess/LCIOIndex to SIO
  - allows to create index of LCIO events over arbitrarily large sets of files
  - direct access to events - possibly w/ pre-selection criteria **(E_t>50GeV)**
- first implementation for Java exists in exp. cvs branch - need to test and implement in C++

partial reading & splitting of events

- needed for performance and cost (disk space) issues:
  - read only objects of interest in analysis (e.g. 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
  - (not yet file splitting)
  - need further testing & implementation in SIO (also Java)
  - need extension of LCIO::Reader interface

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 test beam experiments
- 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 !

ROOT I/O for LCIO

- user request to have closer link of LCIO and 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
  - 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 WG phone meetings for details
- still some issues to resolve (interface to Java !!)

a ROOT dictionary for LCIO

- the latest version of LCIO v01-12-03 allows to optionally create a ROOT dictionary for all LCIO classes - with this one can:
  - use LCIO classes in ROOT macros
  - write simple ROOT trees, e.g. std::vector<MCParticleImpl*>
  - use TTreeDraw for quick interactive analysis of LCObjects:

```
//--- positions of 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

Improving the LCIO event data model

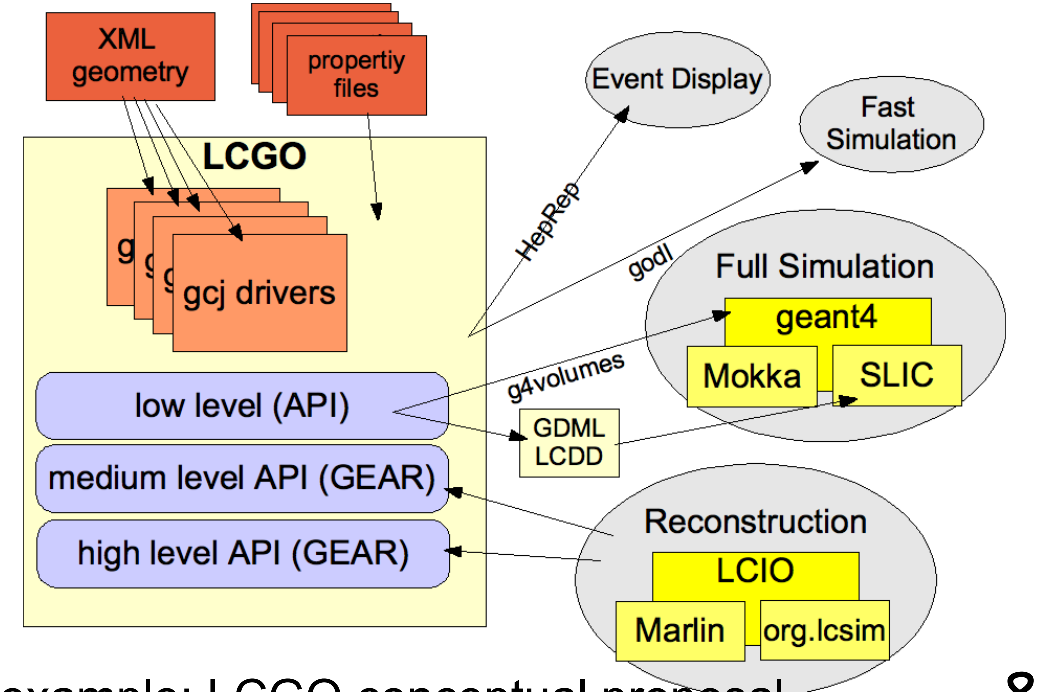
- planned improvements to the event data model:
- 1D, 2D tracker hits
  - LCIO (Sim)TrackerHit is a 3D space point - whereas actual measurements are either 1D (strip) or 2D (TPC) where the detector surface (line) provides the additional geometry information
- Track
  - the current LCIO Track class consists of pointers to all TrackerHits and one set of (Helix) parameters to these hits
  - generally want multiple fits for one set of hits, e.g. at the IP or at the face of the calorimeter
  - could introduce Trajectory as high level convenient view to these fits
  - currently not straight forward (though possible) to store kinks in LCIO
- need close collaboration with and feedback from people working on the new ILD tracking ...

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

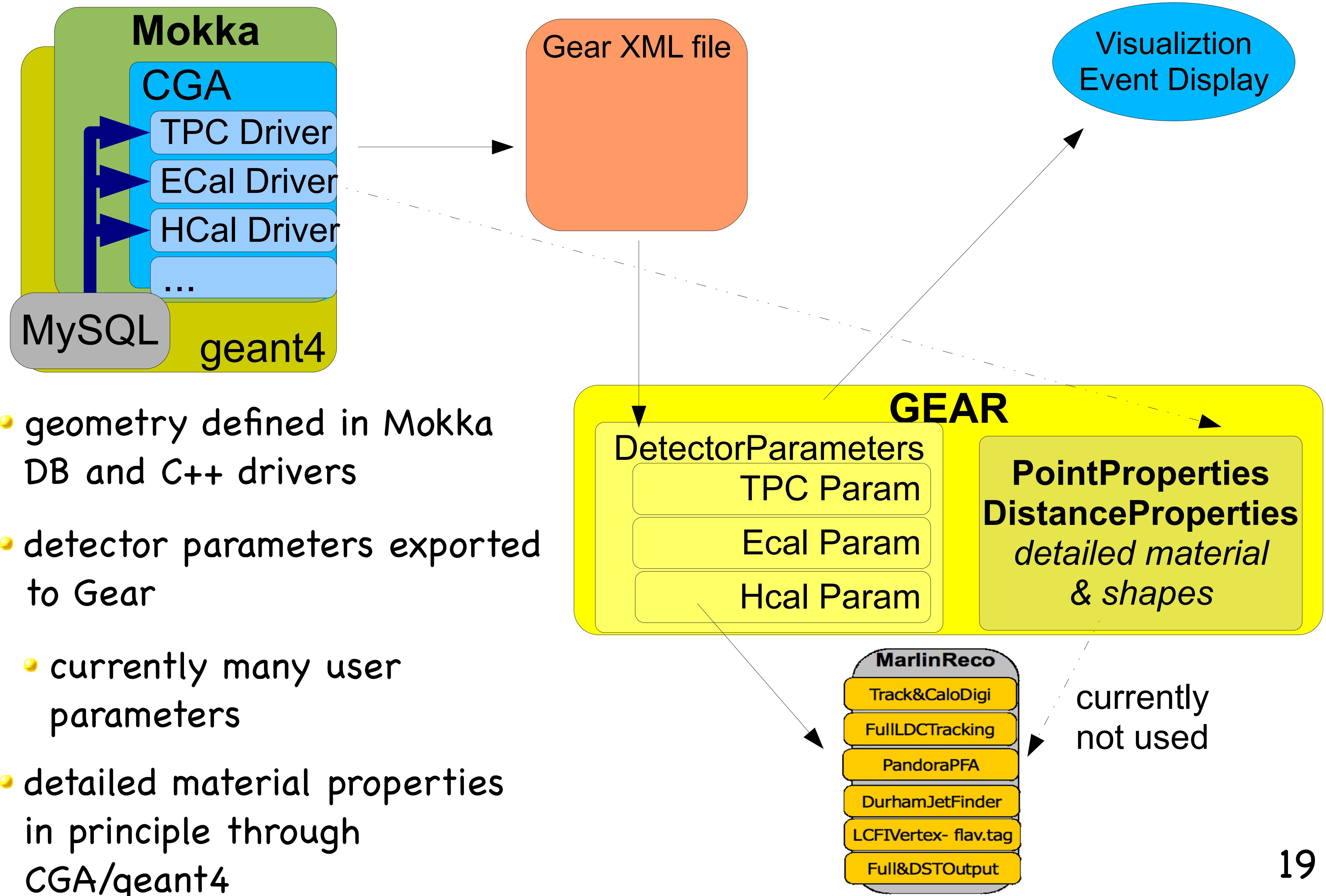
Development of such a toolkit would be part of AIDA fp7 project

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



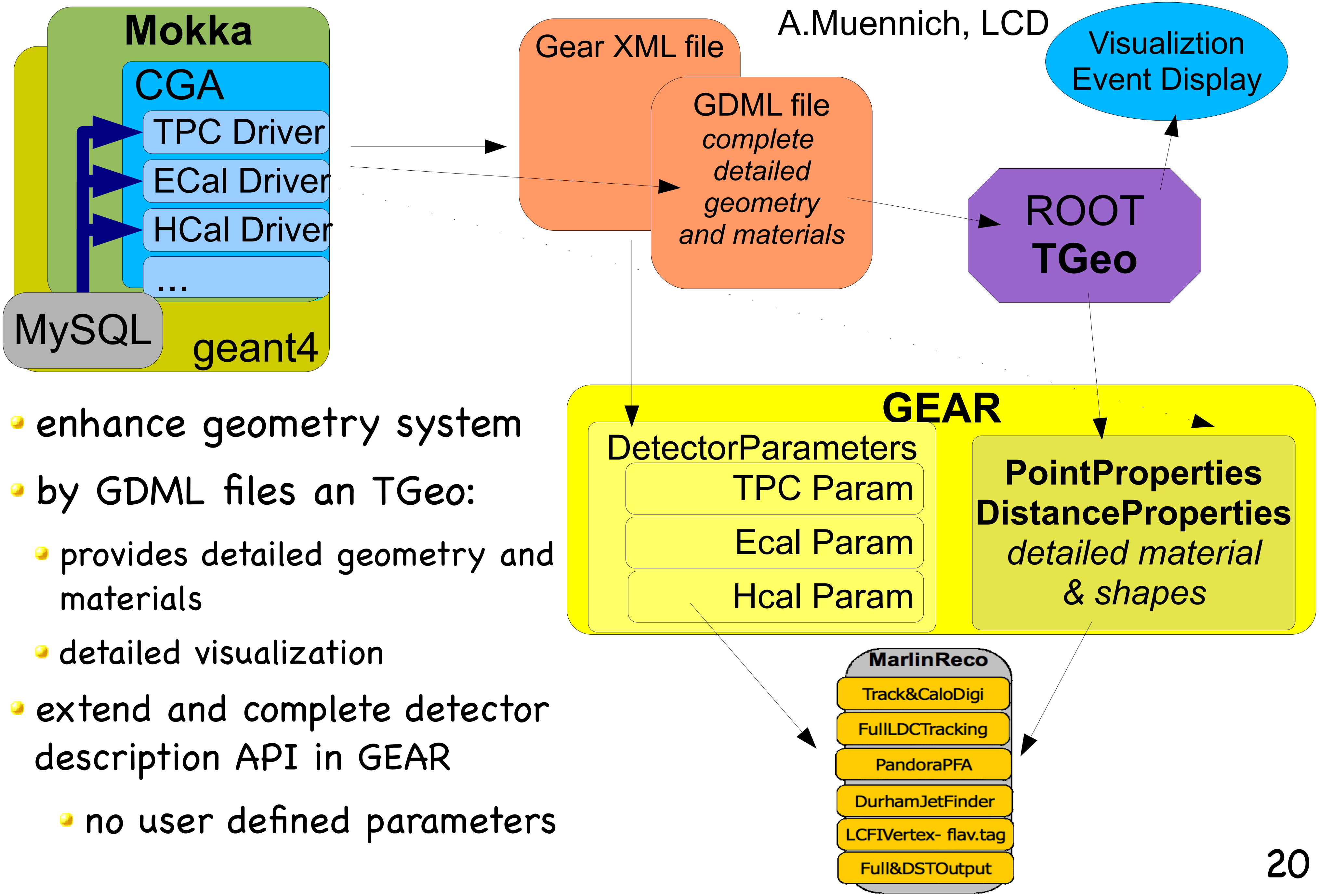
example: LCGO conceptual proposal

current geometry description



- geometry defined in Mokka DB and C++ drivers
- detector parameters exported to Gear
  - currently many user parameters
- detailed material properties in principle through CGA/geant4

improving Gear using GDML & TGeo



- enhance geometry system
- by GDML files an TGeo:
  - provides detailed geometry and materials
  - detailed visualization
- extend and complete detector description API in GEAR
  - no user defined parameters

Summary & Outlook

- ILD has a complete software framework that is battle proven in LOI mass production for detector optimization and physics analyses
- started new phase to further improve the core tools to get ready for the DBD – focus on
  - **LCIO v2**
  - **new improved geometry description**
- need to keep in synch with other developments: realism in simulation, new tracking code, background studies, mass production
- Outlook: proposed EU AIDA project might provide some funding for ILC software development:
  - geomtetry, tracking, particle flow,...