Status of iLCSoft

Overview and Recent Developments

F.Gaede, DESY

AWLC 2017, SLAC, Jun 26-30, 2017

Outline

- Introduction
- Core and Simulation tools
  - DD4hep, lcgeo, ddsim
- Reconstruction tools
  - MarlinTrk, PandoraPFA, HLR
- Migration to GitHub
- Summary and Outlook

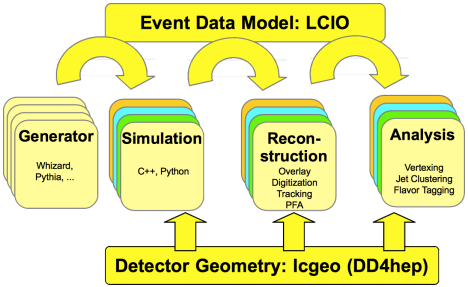
Acknowledgment

People that contributed work and material: M.Frank, D.Jeans, M.Kurata, S.Lu, J.Marshall, N.Nikiforou, M.Petric, A.Sailer, T.Suehara, L.Tran, G.Voutsinas, G.Wilson, ...

Introduction to iLCSoft

- LC community moved to common software tools
  - **DD4hep** detector geometry description
  - **DDG4/ddsim** for simulation
  - **DDRec** as interface for reconstruction
  - new MarlinTrk tracking
  - PandoraPFA
  - LCFIPlus
  - ...
- **iLCSoft now used by ILD, CLICdp and SiD**

note: in this talk most examples are from ILD

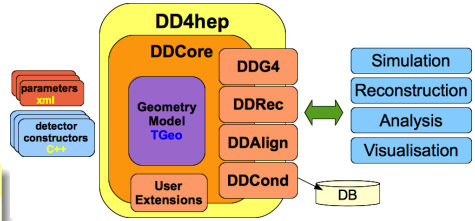


Main recent activities in iLCSoft

- continued the transition from **Mokka/Gear** to **DD4hep/ddsim**
- further improved the complete tool chain
  - core tools: DD4hep, lcgeo,...
  - reconstruction: MarlinTrk, DDMarlinPandora,...
  - High Level Reconstruction tools
- migrated to **ROOT 6**
- use now **c++11** consistently
  - requires gcc49 (or higher)
- moved all major packages to **GitHub**

DD4hep

- DD4hep common detector geometry description tool
  - used by LC,FCC and evaluated at LHCb
- **feature complete** since LCWS 2015



recent activities:

- DDCond added - interface to conditions data
- DDAlign improved
- migrated to ROOT6 and c++11
- implement multi-segmentations
- re-organized the *namespaces*
- many (smaller) issues fixed and new features added

DD4hep - Conditions and Alignment

- DD4hep Mini-workshop at CERN discussed the treatment of *conditions data* and *alignment*
  - reached agreement on how to separate *constant* detector geometry and *changing* alignment in one job
  - will work for **Gaudi** and **Marlin** frameworks
- implications on MarlinTrk/aidaTT:
  - assume that for iLCSoft the alignment does not change during the job
  - can use the *constant* detector geometry with *one* set of alignment constants for tracking geometry (*DDRec/Surfaces*)
  - will implement possibility to load alignment on job startup
- when parallelization will be introduced in **Marlin**, add the possibility to access changing *conditions and alignment data*
  - **after** ILD MC production

DD4hep - namespace changes

- renamed main class: `DD4hep::LCDD` to `dd4hep::Detector`
- renamed namespaces to be more consistent with component structure
- cleanup of some namespaces and deprecated classes

Component	namespace	old namespace
DDCore	`dd4hep`	`DD4hep::Geometry`
DDRec	`dd4hep::rec`	`DD4hep::DDRec`
DDG4	`dd4hep::sim`	`DD4hep::Simulation`
DDCond	`dd4hep::cond`	`DD4hep::Alignment`
DDAlign	`dd4hep::align`	`DD4hep::Conditions`

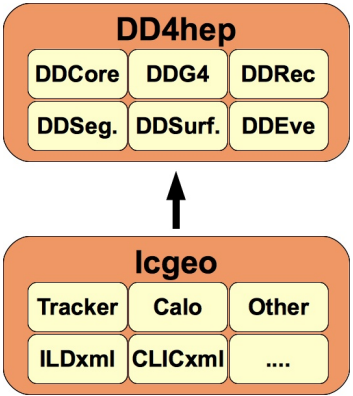
- consistently implemented relevant changes in all iLCSoft packages last week
- preparation of new iLCSoft release ongoing ...

lcgeo

- DD4hep based detector geometry for LC detectors
- used by ILD, CLICdp and SiD, FCal, Calice
- features:
  - mandatory envelope volumes
  - shared detector constructors
  - **ddsim** python executable

recent activities

- added recent models for all concepts:
  - *CLIC_o3_v11, IL4_l1_v02, ILD_s4_v02, SiD_o2_v03,...*
- active development phase with frequent commits



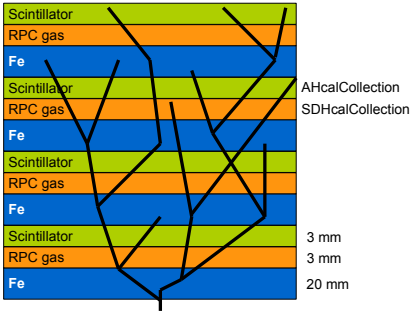
Example: new IID simulation models

large DBD like	new smaller model	technology	Hcal geometry	Ecal
ILD_l1_v02	ILD_s1_v02	analog	Tesla	SiW
ILD_l2_v02	ILD_s2_v02	semi-digital	Videau	SiW
ILD_l4_v02	ILD_s4_v02	*both*	Tesla	SiW

- defined a number of new ILD simulation model with different calorimeter technologies
- one **large** model (a la DBD) and a new **smaller** model (a la CLIC) each
- will use *multi-technology* readout models for the planned large MC production

multi-technology simulation model for the ILD Hcal

- calorimeter shower development basically defined by absorber structure
- created simulation models with two sensitive materials:
  - scintillator and RPC gas
  - *same thickness* as in individual models
- will us this in large scale MC production
- possibility to **compare technologies** on full physics analysis using the **same events**
- use new DD4hep feature of *multi-segmentations*



Multi-Segmentations in DD4hep

- introduced the possibility to have multiple segmentations and/or collections within one readout
  - the correct one is selected based on a *key* of the *cellID*, e.g the *slice*:

```
<readout name="HcalBarrelReadout">
  <segmentation   type="MultiSegmentation"  key="slice">
    <segmentation name="RPCgrid" type="CartesianGridXY" key_value="2"  grid_size_x="SDHCal_cel
    <segmentation name="Scigrid" type="TiledLayerGridXY" key_value="4"  grid_size_x="3" grid_s
  </segmentation>
  <hits_collections>
    <hits_collection name="HCalBarrelRPCHits"  key="slice" key_value="2"/>
    <hits_collection name="HcalBarrelRegCollection"  key="slice" key_value="4"/>
  </hits_collections>
  <id>system:5,module:3,stave:4,tower:5,layer:6,slice:4,x:32:-16,y:-16</id>
</readout>
```

- here the slices 2 and 4 are sensitive with RPC-gas and scintillator, respectively
  - also defined as sensitive in compact file

ddsim

- python application for running full simulation
- uses DDG4 component of DD4hep
- easy to use with steering files or command line options to configure:
  - sensitive detectors
  - filters
  - cuts
  - ...
- for all options simply run

```
ddsim --help
```

status

ddsim is fully functional and already used for (test) productions

example call:

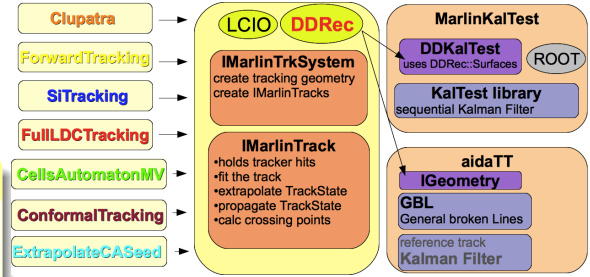
```
ddsim --inputFiles ./bbudsc_3evt.stdhep \
--outputFile ./bbudsc_3evt.slcio \
--compactFile $lcgeo_DIR/ILD/compact/\
ILD_l1_v01/ILD_l1_v01.xml \
--steeringFile ./ddsim_steer.py
```

MarlinTrk

- generic track fitting toolkit
- modified to use *DDRec::Surface*
  - automatic materials from detailed model
- many different patrec codes exist

recent developments

- complete tracking software chain - including digitizers - is now **Gear-free** and uses **DD4hep/DDRec only**

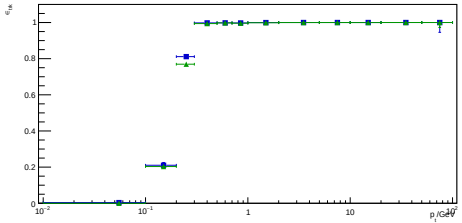


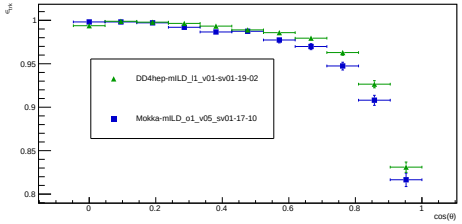
interface to fitters for pattern recognition codes

Example: tracking efficiency with DD4hep for ILD model

- comparison of tracking efficiency large ILD model:
  - based on Mokka simulation
  - using complete new DD4hep chain (DDKalTest, MarlinTrk)
- *observe issues:* at 90° and for low p_t
  - under investigation

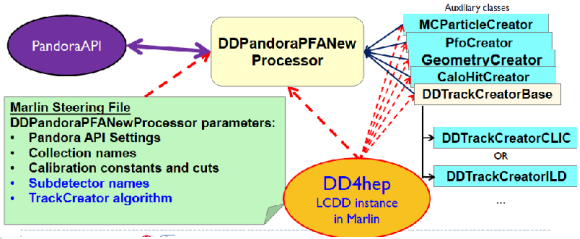
- new DD4hep based tracking reaches better tracking efficiency than with *old chain* almost everywhere
  - small issues under investigation





PandoraPFA - DDMarlinPandora

- **PandoraPFA**: generic particle flow toolkit used by LC and neutrino communities
- **DDMarlinPandora**: interface for between DD4hp/Marlin and the *PandoraSDK*



recent activities

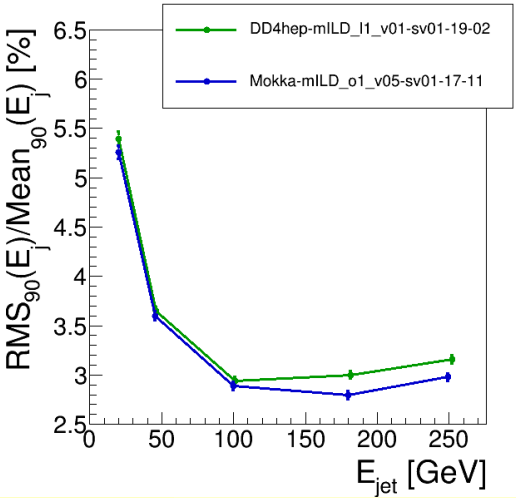
- improved photon clustering
- software compensation
- DDMarlinPandora calorimeter description improved
- uses **c++11** only now

alternative PFA code: **Arbor** under development

Example: JER performance with DD4hep for ILD

- comparison of JER for uds events for large ILD model:
  - based on Mokka simulation, using latest PandoraPFA (w/ SW-compensation)
  - using complete new DD4hep chain
- *known issue:* need proper calibration and re-tuning of sw-compensation for new chain

- complete new DD4hep based software chain reaches (almost) the performance achieved with the *old* chain
- note: no GEAR is used up to the creation of PFOs any more



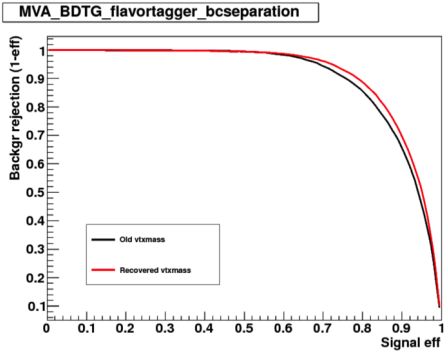
LCFIPlus

- flavor tagging and vertex finding tools

recent developments

- adaptive vertex fitting
  - assign leftover tracks to vertices
- vertex mass recovery
  - assign π^0s to vertices
- fixed issue in TMVA in ROOT 6.08
  - use ROOT ≤ 5.29 or ≥ 6.08

- potential issue w/ crash under ROOT 6 is under investigation



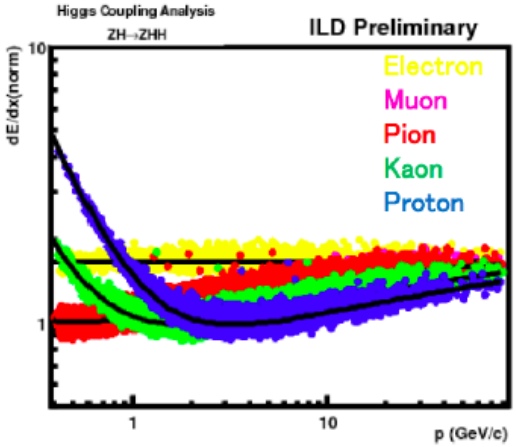
improvement of flavor tagging w/ vertex mass recovery

Other High Level Reconstruction Tools

quite a number of packages have been added or extended, e.g:

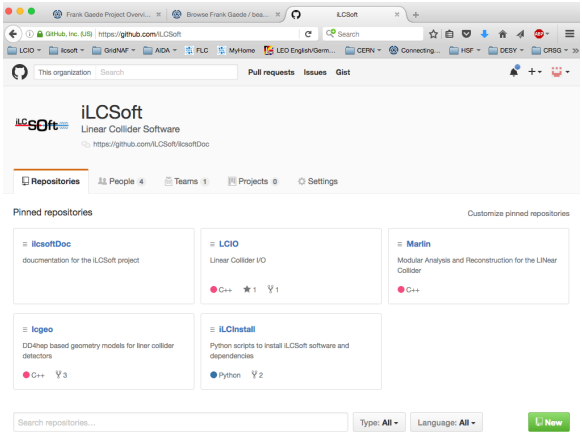
- *PIDTools:* improved Particle-ID (dE/dx (TPC), shower shapes and MVA)
- *GammaGammaCandidateFinder:* new π_0 finder
- *RealisticCaloDigi:* new DD4hep compatible calo digitizers
- *MarlinKinfit:* improved fitting with track objects

- **To Do:** need to replace Gear with DD4hep/DDRec in HLR tools



Transition of iLCSoft to GitHub

- at Software Meeting in Santander (June, 2016) decided to move to **GitHub** as main code repository for iLCSoft
- set up iLCSoft organization at `https://github.com/iLCSoft`
  - with all iLCSoft packages from svnsrv.desy.de
- at Software Hands-On-Week at CERN in March **finalized** the transition from SVN to GitHub
- all *core iLCSoft* packages are maintained and developed in GitHub since



Some Technicalities of transition to GitHub

- in transition made sure all packages have a
  - `LICENSE` file
    - if they use *Marlin* we have added: **GPLv3**
  - a `README.md` with
    - basic introduction to package
    - **copyright** statement, e.g.
    - copyright: the *package_name* authors
  - `AUTHORS` file
    - listing authors that have made *significant* contributions
  - ./doc/ReleaseNotes.md

let us know if any of these files are not correct or incomplete

- in particular the AUTHORS file

Implications of transition to GitHub

- everyone contributing to iLCSoft needs to get a **GitHub account**
  - at `https://github.com/join` - using their **real name**

basic GitHub workflow

- create a fork of the package repository
- make your changes in a dedicated *feature branch*
- commit (push) to your private fork of the repository
- create a Pull Request (PR) on the GitHub page
- experts will review your changes and eventually merge them

- see `https://github.com/iLCSoft/ilcsoftDoc` for details
- and **GitHub tutorial** given at ILD workshop in Lyon:

`https://agenda.linearcollider.org/event/7520/contributions/38807/attachments/31483/47412/170424_GitTutorial.pdf`

Why move to GitHub ?

Review Mechanism

- everyone can review and comment on PRs
- no experimental or sloppy code gets merged into the main repository
  - users can push this to their own forked repositories

Continuous Integration:

- every PR starts *builds and test* for different compilers
  - only if these are successful the PR will be merged

Static Code checking

- set up *Coverity* for all packages -> finds *logical flaws* in code

users and account management

- done by users via GitHub

Summary and Outlook

Summary

- iLCSoft tools now shared by all LC concepts
- have completed the transition from the old *Mokka/Gear* world to the new *DD4hep/lcgeo* based one
- completed migration to **ROOT6**, **C++11** and **GitHub**
- have now a *Gear-free* reconstruction chain for ILD and CLIC, up to the PandoraPFOs
  - need to migrate HLR tools

Outlook

- make a new iLCSoft release v01-19-03 soon (this week)
- continue to address *open issues*
- target a *production* release, later this summer
  - ILD and CLICdp are planning large scale Monte Carlo productions with this