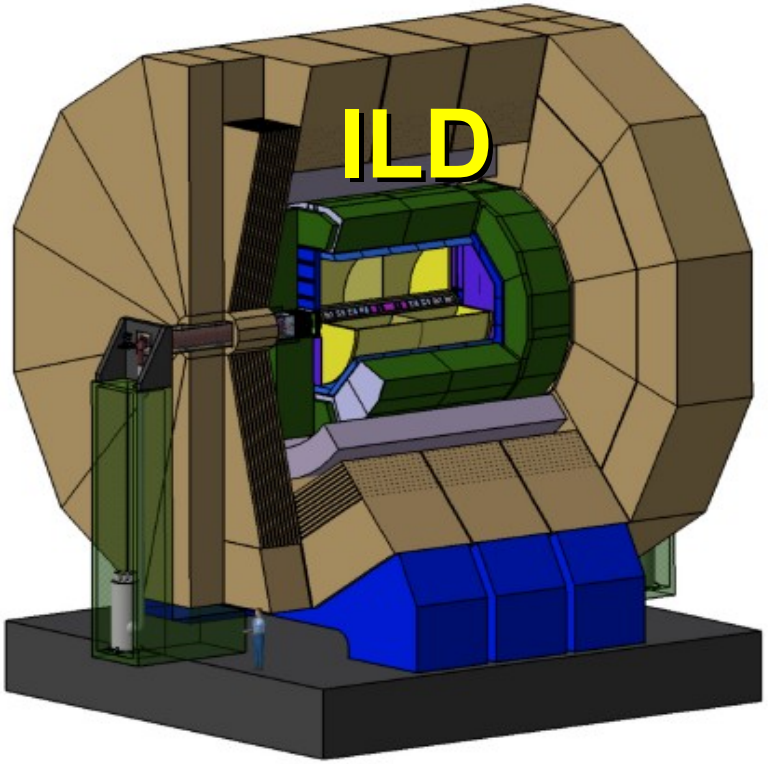
ILD Software

Status and Plans – towards the TDR 2012

Frank Gaede
DESY

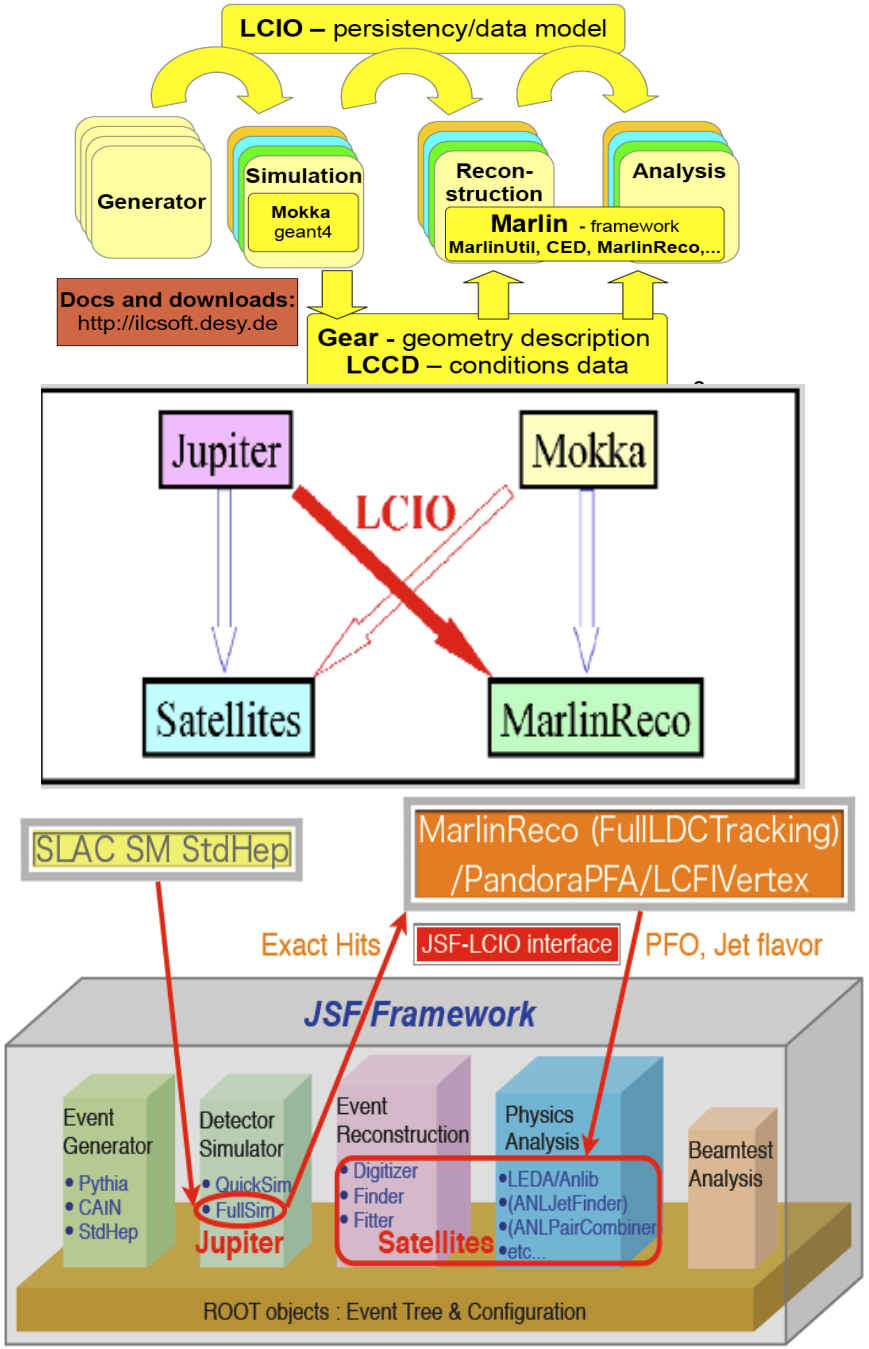
ILD Meeting
Albuquerque, October 1, 2009

Outline



- Overview & Status
- Plans
  - framework
  - simulation, geometry, reconstruction
  - LCIOv2
- Timeline
- Summary & Outlook

ILD Software Introduction

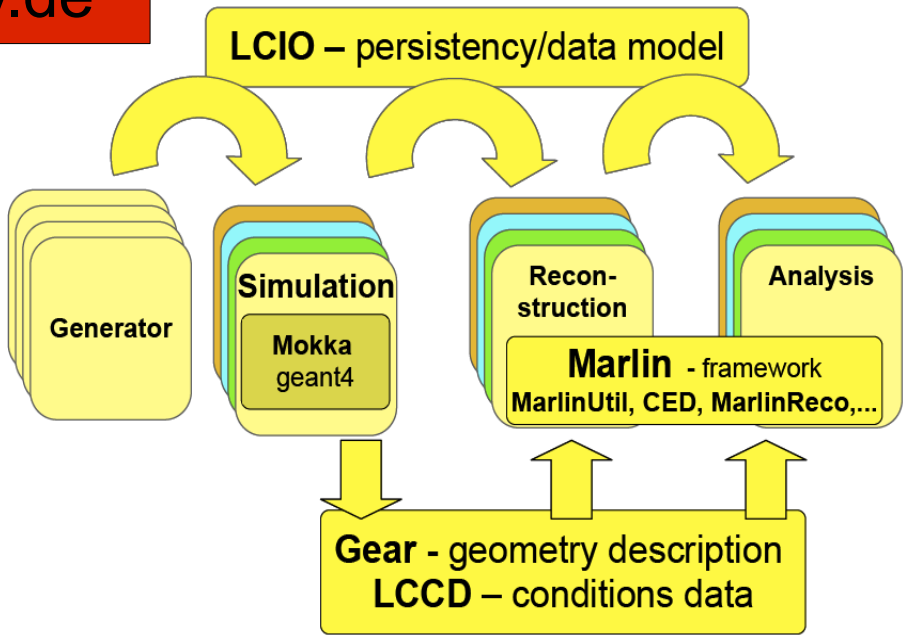


- ILD had two frameworks at beginning of LOI phase
- both frameworks 'battle proven' in massive Monte Carlo production
- interoperability provided by common usage of LCIO
- started to move towards a common ILD software framework – based on Marlin, LCIO with 'Goddies from Jupiter&Satellites

ILD Core Software Tools

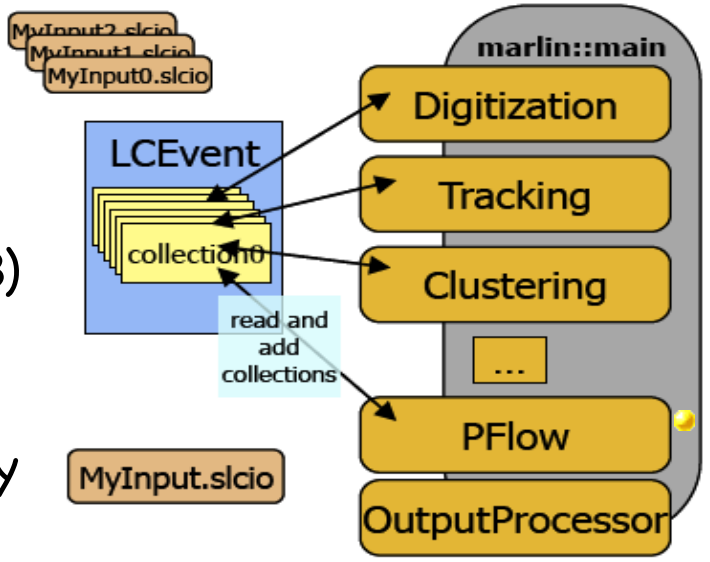
http://ilcsoft.desy.de

- **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



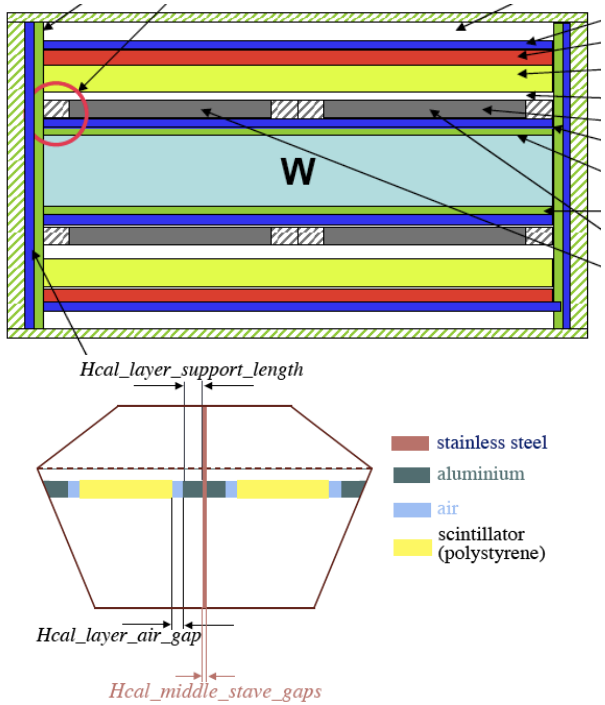
- complete framework used in Monte Carlo & 'real experiments':
  - **ILD detector concept** studies
  - **Calice** calo testbeam
  - **LC-TPC** testbeam
  - EUDET - **Pixel Telescope**

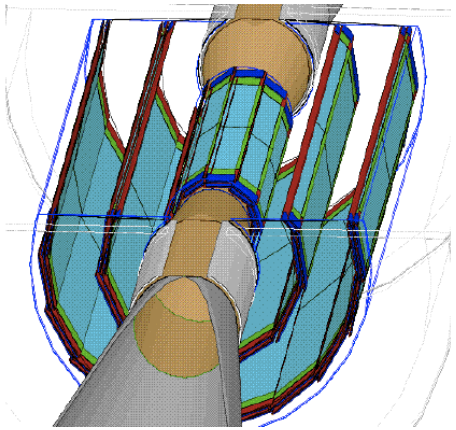
synergies between testbeam and global detector optimization

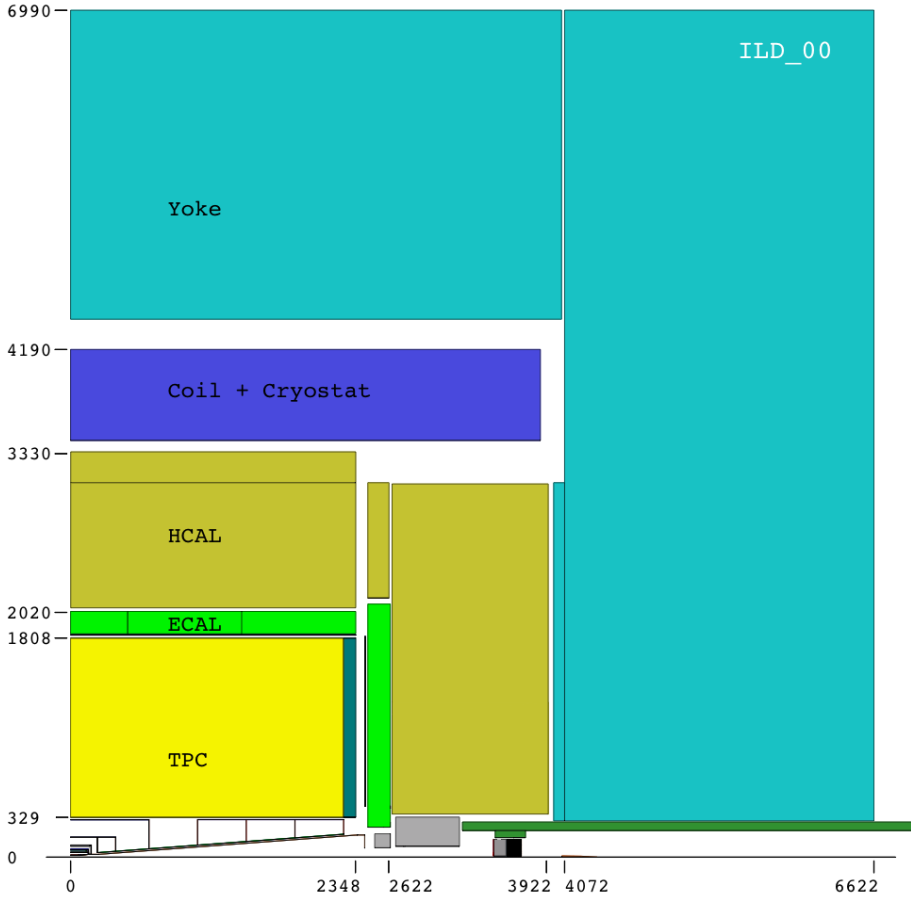


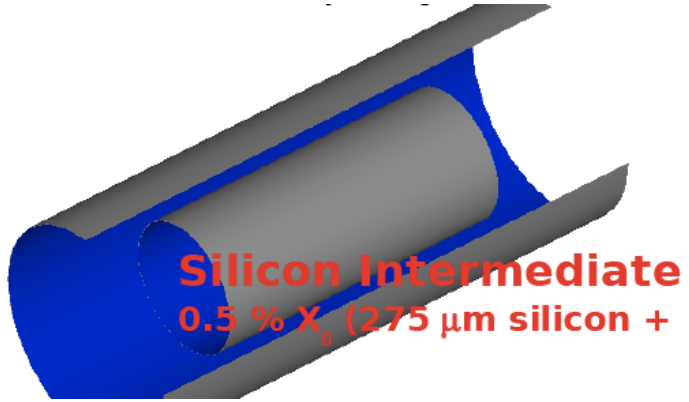
Mokka Simulation

- defined 'ILD simulation reference model' after LOI optimization process LDC and GLD
- engineering level of detail for most subdetectors:
  - support structures
  - dead material (cabling, cooling)
  - cracks









Marlin based Reconstruction

- Tracking
  - standalone tracking in Silicon detectors and TPC
  - 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
  - now maintained and improved by japanese groups !
- DST Maker
  - ReconstructedParticles, Jets, Tracks and Clusters (25k/evt)
- Overlay
  - overlay of fully simulated background events of N bunch crossings

- software was used quite successfully for the LOI:

> *"The ILD efforts on simulating the physics benchmark processes have been impressive."*
> **IDAG Report on the Validation of Letters of Intent for ILC detectors**

- however,
  after the LOI is only before the TDR ...

preparing the ild software for the TDR

Guideline for the Plan of the detector groups

1. **Continue R&Ds on critical components to demonstrate proof of principle**
2. **Define a feasible baseline design**
   (Options may also be considered. But one of them should be proven to be feasible.)
3. **Complete basic mechanical integration of the baseline design accounting for insensitive zone**
   (such as support structure, pipes, power lines etc.)
4. **Develop a realistic simulation model of the baseline design, including faults and limitations**

Guideline (cont'ed)

a push-pull mechanism working with relevant

a realistic concept of integration with the accelerator including the IR design

5&6: with GDE's BDS group through the MDI group

7. **Simulate and analyze benchmark reactions, which can be updated**
8. **Simulate and analyze some reactions at 1 TeV, including realistic higher energy backgrounds demonstrating the detector performance.**

8&9: Based on the work of the Physics Group and Software group.

The reaction will be chosen to show the strength of ILC compared to other facilities.

Plans for ILD software

- at Cambridge decided to move towards common ILD software framework – developed plans in
- (bi)weekly ild-sw-mgmt meetings
- dedicated ILD software workshop in Tsukuba 2009
- bi-weekly software WG meetings:
  - 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

Merge goodies from JSF in to ildsoft

- port useful features of the core framework such as command line options to Marlin
- adopt selected JSF modules to be run as Marlin processors, e.g. the QuickSim fast simulator
- port subdetector simulation code from Jupiter to Mokka for technologies that are not present in Mokka, such as the Scint. ECal
- make existing analysis code from JSF available and compatible with LCIO (possibly via ROOT dictionaries)

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,...
- comment: such a test systems would probably have saved us some hassle in the past - and will make future development more efficient !
- need to be pragmatic about this: look into existing testing tools and/or extend our installation toolkit

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)

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
- ...

ideally collaborate with other HEP groups on that !

Improve Reconstruction Tools

- digitization:
  - improve description of spacial resolution (R&D groups)
  - introduce ghost hits for strip detectors
- tracking:
  - develop modern tracking and pattern recognition software to replace f77 LEPTracking
    - proper treatment of strip detectors
    - tracking in non-uniform B field (anti-DID)
- clustering/PFA
  - modularize and improve PandoraPFA

needed for proper background studies !

LCIOv2

- improve LCIO -> LCIOv2
- event data model
  - 1d, 2d hits
  - Track class - multiple fits per track
- Improve I/O
  - splitting of files
  - direct access
  - partial reading of events
- investigate the use of ROOT with LCIO
  - LCEvent in ROOT macros
  - look into optional ROOT I/O for LCIO

continue successful horizontal collaboration with SID on LCIO

ROOT I/O for LCIO

- started to investigate optional ROOT I/O for LCIO
- created dictionary with rootcint for LCIO classes
  - thanks to ROOT team for their help and for adding some features to ROOT 5.24.00 needed for LCIO
  - => write and read LCEvents transparently to/from ROOT files
    - no change in user code !
  - => use LCEvents in ROOT macros
    - rapid development of analysis code based with LCIO in ROOT !
- issues:
  - no branches due to pointers between object
  - no partial reading and splitting of events over files
  - need proper interface to ROOT I/O for java implementation

-> need to work with ROOT team to resolve these issues...

Extend and Improve Simulation

- need '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)

improving the simulation

- almost all of the Mokka subdector drivers have been written by experts from the R&D groups
  - this is of course the right approach as it ensures that 'state of the art' sub detector description is used
- ideally it should also be the detector R&D groups that maintain the simulation code, ie. test, debug and further improve it

- would like to have one name per subdetector that is responsible for the simulation and serves as contact to the core software group !

what level of realism is needed ?

- Mokka simulation already quite realistic for some detectors – clearly need to improve for others ?
- how realistic do we have to be ?
  - describing every cable and every nut and bolt is neither feasible nor necessary
  - need to be realistic with cracks and clearances (for cable channels etc.)
  - need to get good estimate from R&D groups/ MDI group on cable/support material budget
- in any case what eventually can be done depends on the effort in terms of manpower
- and the discussion within ILD !

timeline for ILD software development

t0	hand in TDR	13 month
5 month	Analysis and Writing	
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 month	implement baseline in simulation	14-26 month
t0-x	ILD baseline defined	
	evaluate technology options develop tracking package develop geometry LCIOv2 improve simulation realism improve reconstruction study machine backgrounds ...	

13 month prior to hand in of TDR are **fixed**

depending on actual t0 we have ~1-2 years for software developments...
e.g. t0=Oct 2012
=> 23 months from now !

Summary

- ILD has a complete software framework that is battle proven in LOI mass production for detector optimization and physics analyses
- now entered new phase:
  - merging the two frameworks into one common ILD framework
  - further improve the tools to get ready for the TDR 2012
- already have quite ambitious plans for software development – clearly need support from ILD
- in the end what can be done realistically depends on the TDR-t0 and on the effort that we put into the software development