ILD Software
Overview, Status and Plans

Frank Gaede
DESY
2009 Linear Collider Workshop of the Americas
Albuquerque, September 30, 2009
Outline

• Introduction

• Overview
  • install tools
  • simulation
  • reconstruction

• Plans
  • framework
  • simulation, geometry, tracking, PFA
  • LCIOv2

• Summary
ILD Software Introduction

- ILD had two frameworks at beginning of LOI phase
- both frameworks 'battle proven' in massive Monte Carlo production
- see Steve's talk for details and 'lessons learned'
- started to move towards a common ILD software framework – based on Marlin, LCIO with 'Goddies from Jupiter&Satellites
ILD Core Software Tools

- **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

**http://ilcsoft.desy.de**

**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
ILD software builds and installation

`ilcinstall` tool: python scripts to download, build and install all ILD and external packages (incl. Jupiter&Satellites)

- '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)
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

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

- MarlinReco processors for digitization:
  - VXD, SIT, FTD, SET, ETD Silicon hits
    - smearing of 3d space points (SimTrackerHits) according to envisaged detector resolutions
      - as established by R&D groups
  - TPC hits
    - smearing of 3d space points (SimTrackerHits) 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)
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

- **DST Maker**
  - ReconstructedParticles, Jets, Tracks and Clusters (25k/evt)

StandardConfig package has reference steering file for Marlin
Some Recent Developments

- improved CED event display
- fish eye view
- picking (under development)

- improved tracking code for background studies
  - Overlay processors
  - silicion digitizers (cluster sizes)
  - TPC bunch train integration of machine backgrounds
Plans for ILD software

- after LOI it is time to further improve ILD software and get ready for TDR phase (2012)
- plan to:
  - merge goodies from JSF into framework
  - develop a test system
  - develop new GRID production system
  - improve the simulation
  - improve the geometry description
  - improve the reconstruction (tracking & PFA)
  - develop LCIOv2
Merge goodies from JSF into 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)
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)
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 spatial resolution (R&D groups)
  - 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)

- **clustering/PFA**
  - modularize and improve PandoraPFA
LCIO: persistency & event data model

- joined DESY and SLAC project
  - first presented @ CHEP 2003
- provides persistency (I/O) and an event data model (EDM) to ILC detector R&D community

features:
- Object I/O (w/ pointer chasing)
- schema evolution
- compressed records
- hierarchical data model
- decoupled from I/O by interfaces
- C++, Java (and Fortran)
- some generic user object I/O

LCIO is used by ILD, SID, Calice, EUPixelTelescope, LCTPC,...
LCIOv2

- further 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...
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:

More realism for some detectors – additional technologies.

Develop a new and modern tracking package.

Develop a geometry description.

Further improve the digitization and reconstruction algorithms.