Linear Collider - Concurrency Needs and Plans

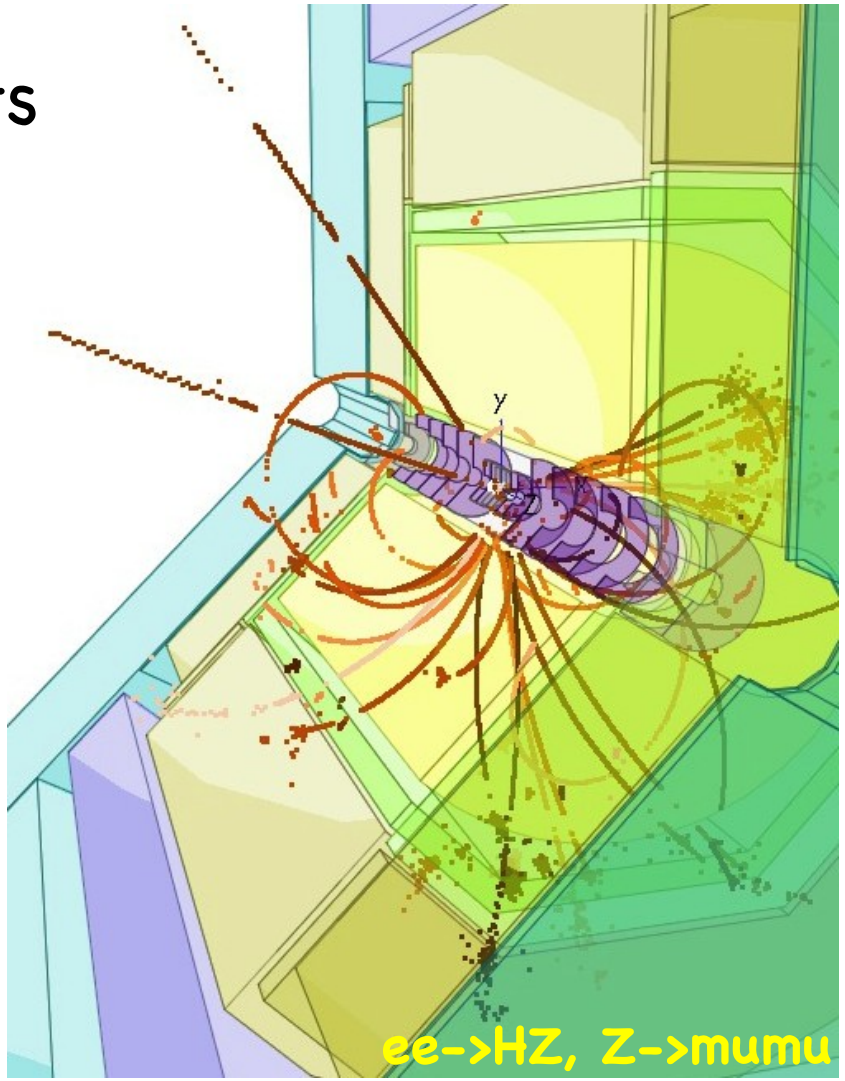
Frank Gaede, DESY

Annual Concurrency Meeting

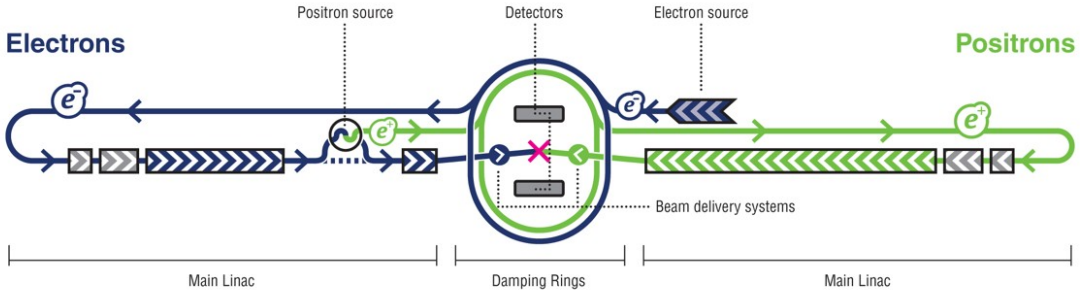
FNAL, Feb 04-07, 2013

Outline

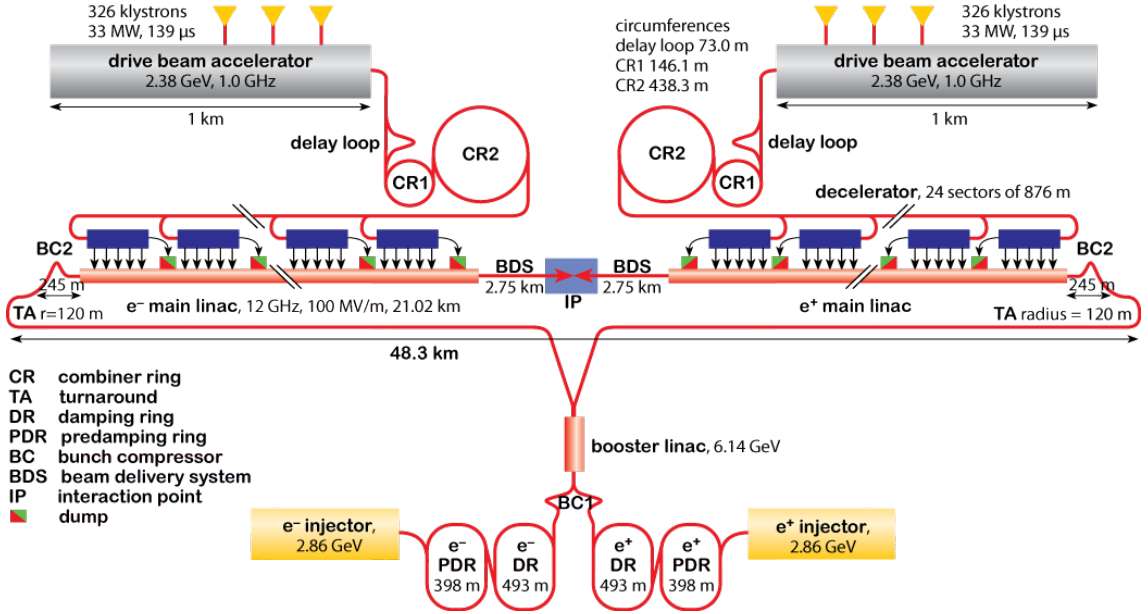
- Intro: Linear Collider Projects
- Software Frameworks
- Monte Carlo Production
- Plans for Concurrency
- Summary & Outlook



Linear Collider projects

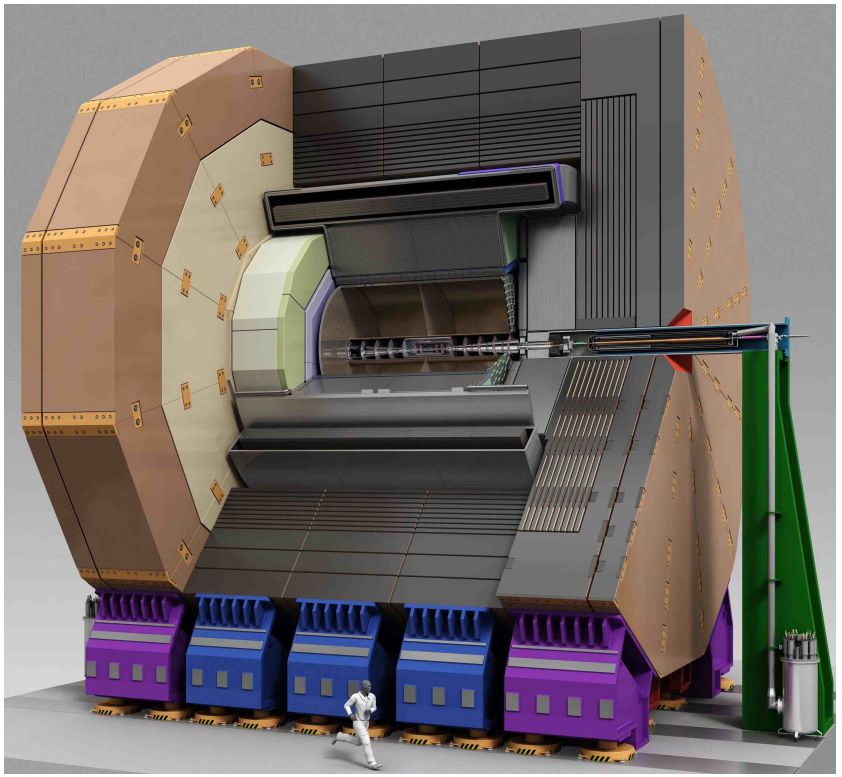


- **ILC**: 500GeV-1TeV
- super conducting RF
- **TDR** to be submitted
  - strong interest in ILC in Japan



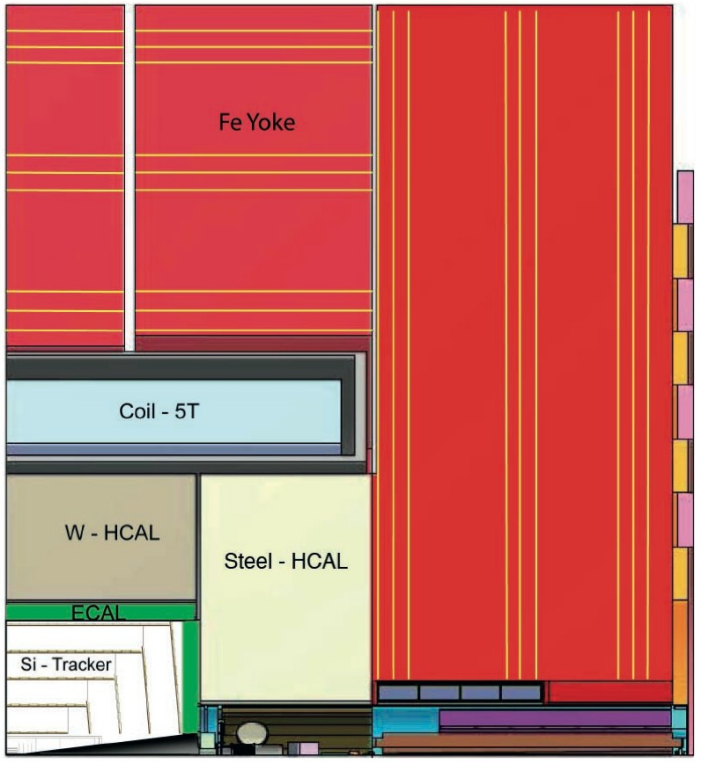
- **CLIC**: 500GeV-3TeV
- drive beam acceleration
- **CDR** submitted
  - possible successor to LHC

Linear Collider detectors

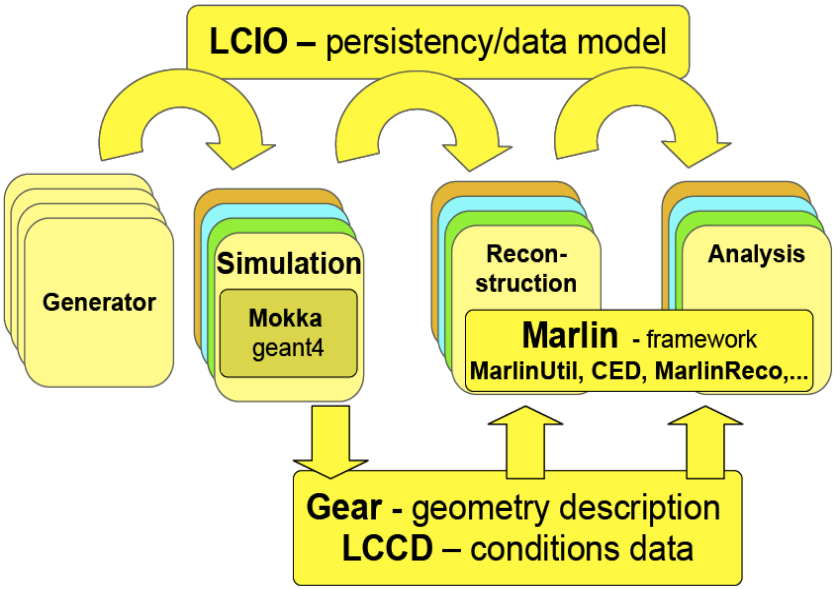


- two ILC detector concepts:
  - **ILD**: 3.5T, TPC
  - **SID**: 5T, Si-Tracker
- both detector concepts also used for CLIC
  - w/ adaptations

- ILC and CLIC detectors are optimized for Particle Flow:
  - precision tracking and vertexing
    - d(1/pt) -> 2.*10e-5
    - d(D0) -> 2-3 mu
  - very high granularity in calorimeters: 1-3 cm in HCal

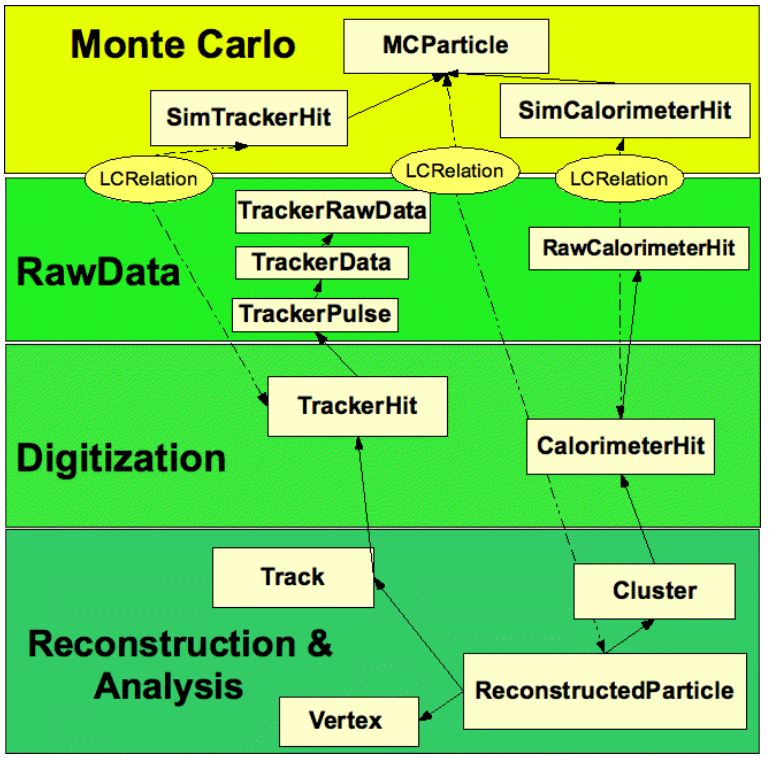


Linear Collider software frameworks



- two frameworks are currently in use for LC software studies
- **iLCSoft**: (LCIO, Mokka, Marlin) C++ framework - ILD, CLIC
- **org.lcsim**: (LCIO, SLIC, org.lcsim) C++ and Java (reco) framework - SID, CLIC

- **LCIO**: common EDM and persistency
  - hierarchical event data model
  - C++, Java and Fortran API
  - machine independent non-ROOT format
  - since 2003 (!),
- LCIO basis for common tools, e.g.:
  - Marlin, PandoraPFA, LCFI+,...
  - goal: common geant4 simulation



LC Monte Carlo Production

- several large scale Monte Carlo productions in recent years for ILD, SID and CLIC: LOIs, CDR, DBD/TDR,...
- typically O(10e8) events in full (geant4) simulation and reconstruction
- some benchmarks (ILD @ILC 1TeV):
  - sim: 5-9 min / event
  - rec: 30-60 sec / event * (w/o bg)
  - rec: 45-210 sec / event * (w/ bg)
  - (numbers for CLIC larger !)
- mostly done on LCG grid infrastructure in VO: ILC
- **computing needs small compared to LHC - dominated by simulation !**

ILC @ 1TeV:
4.1 gg->had events/BX * 1 BX
CLIC @ 3 TeV
3.1 gg->had evts/BX * 60 BX

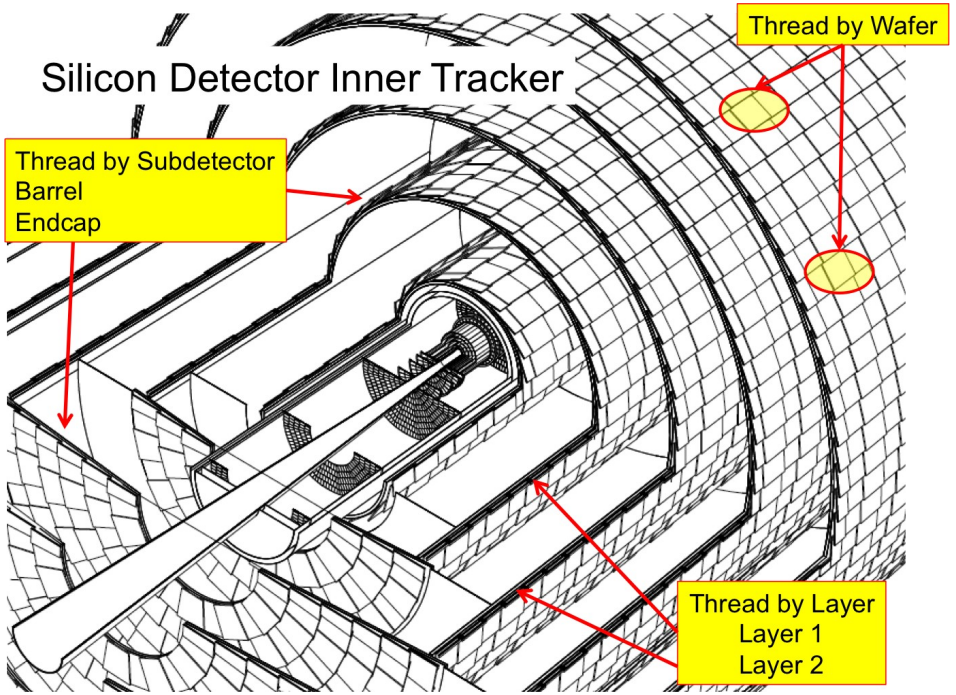
LC need for parallelization

- LC computing needs in general modest compared to LHC
- there is no immediate/urgent need for performance gains through concurrency - however
  - this might change in not so distant future as more and more many core machines are deployed in the Grid
  - of course one should be as efficient as possible with CPU resources

- would benefit most from improvement for full simulation
- -> very interested in trying geant4-MT and following development in geant vector prototype

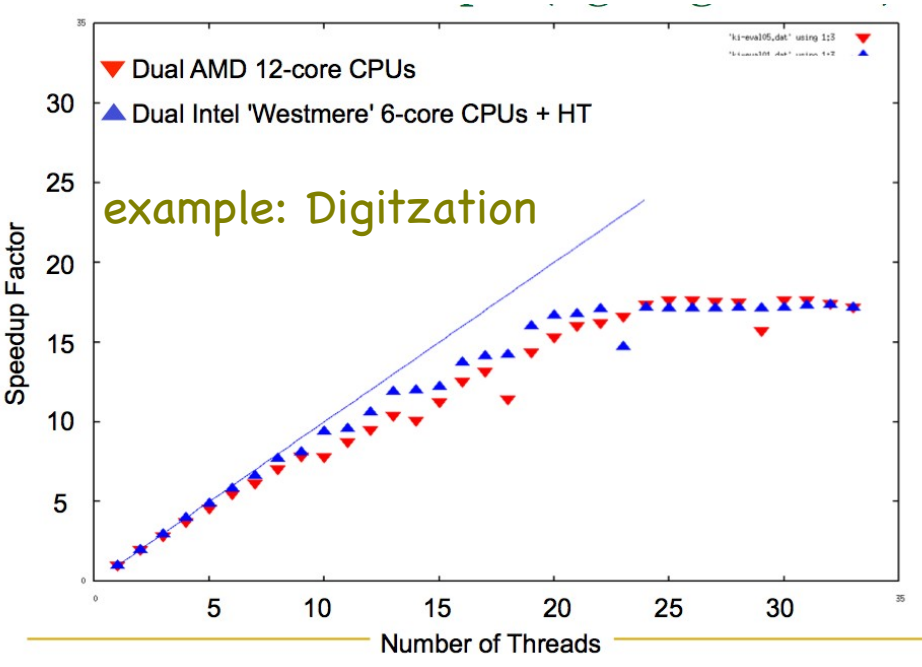
- also some ideas/plans wrt. parallelization between and within algorithms:

example: Java multithreading for LC



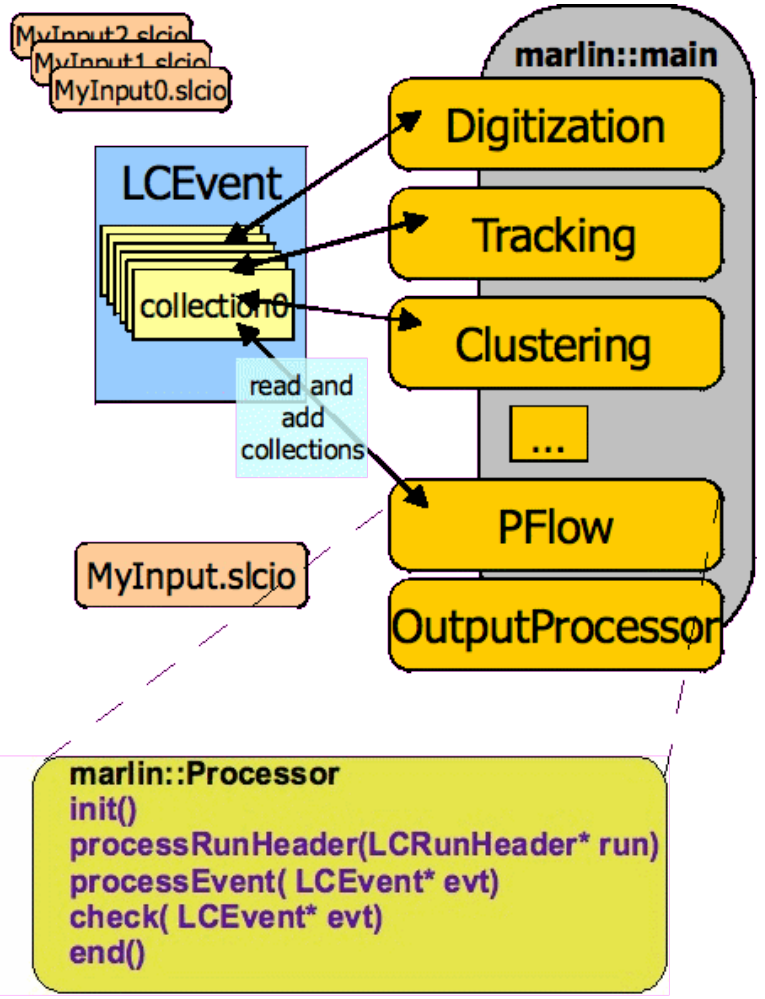
- Java has built in support for multithreading
  - Thread, Runnable, Callable
  - ExecutorService
  - thread safe collections
- new C++-11 will also provide native thread support

- N.Graf: CHEP 2010 Taipei: Multi-Threaded Event Reconstruction in Java
- LC collider (SID) digitization and tracking
- proof-of-concept study
- parallelization between and within the algorithm

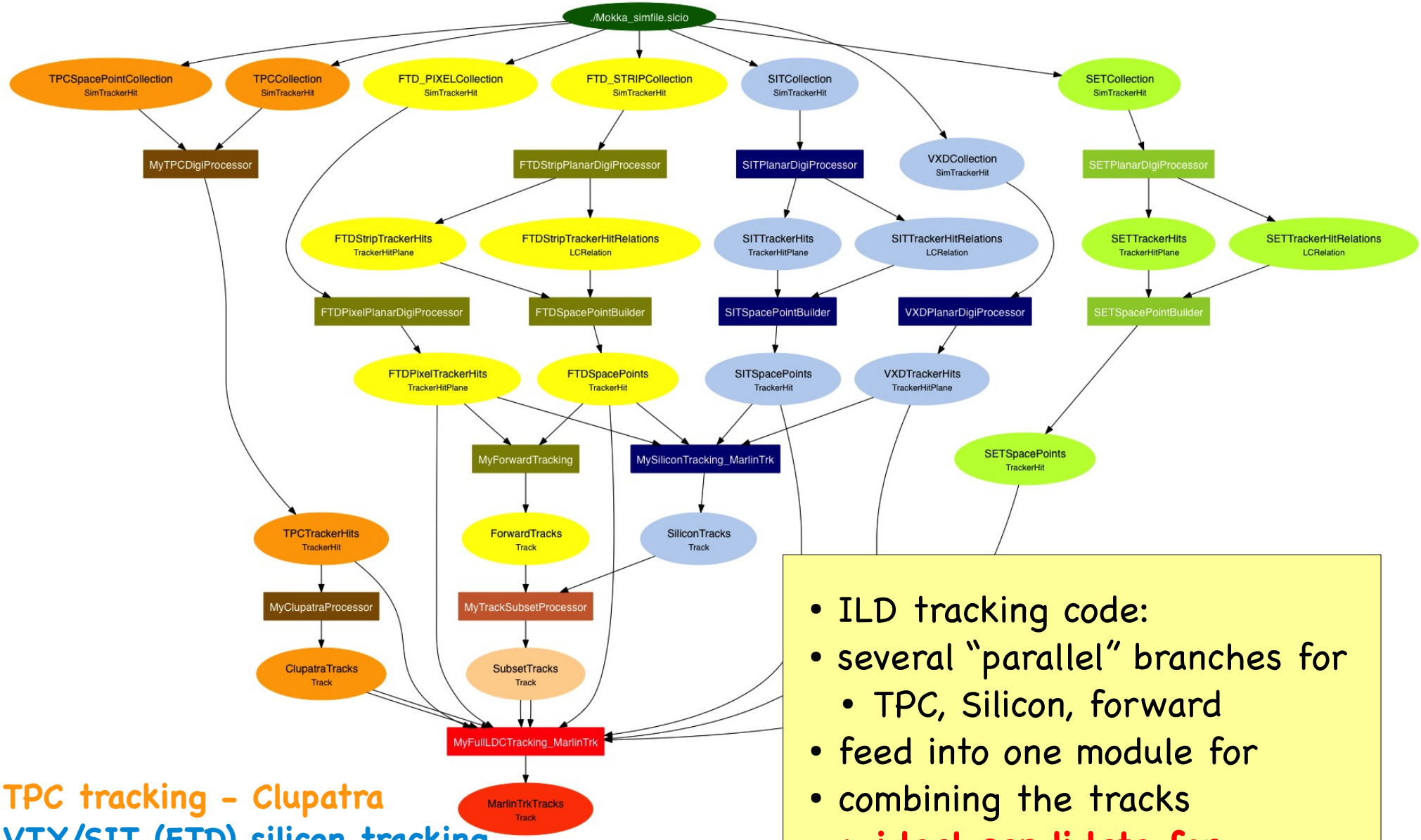


Parallel version of Marlin

- Marlin: modular C++ application framework for LC reco/ana
  - most LC reco code in Marlin modules
  - ideal for parallelization on the algorithm level a la GaudiHive, CMSSW - possibly almost transparent to the users
- LCIO as transient EDM (event bus)
  - (input) collections are read-only
  - event can only be extended
  - -> locking for multithreading should be straight forward
  - -> to be addressed: white board with multiple events (currently only one/reader)
- timescale depends on manpower...



example: ILD tracking in Marlin



TPC tracking - Clupatra

VTX/SIT (FTD) silicon tracking

Forward tracking

FullLDC combined tracking (+SET)

- ILD tracking code:
- several "parallel" branches for
  - TPC, Silicon, forward
- feed into one module for
- combining the tracks
- -> ideal candidate for concurrency on module level

Frank Gaede, Concurrency Forum, FNAL, Feb 04-07, 2013

Summary & Outlook

- two LC software frameworks sharing common EDM and an increasingly large number of software tools used by three groups: ILD, SID and CLIC
- LC computing needs are in general modest compared to LHC
- no immediate/urgent need for performance gains through concurrency
- however interest in following general trend towards concurrency in HEP
  - would immediately benefit from improvement for full simulation

- **Outlook**
- plan to start with module level parallelization in Marlin and follow more closely the Concurrency Forum activities