ILC Software: Where are we?

Ties Behnke, DESY

Software: where are we

where are we going in the future

Software is always difficult to discuss

many personal likes and dis-likes

many clear and often very vocal opinions

What does “core software” mean for a project which is not yet a project

ILC Software: Where are we?

Ties Behnke, DESY

Software: where are we

where are we going in the future

Software is always difficult to discuss

many personal likes and dis-likes

many clear and often very vocal opinions

I present a Europe centric view (ECFA centric)

What does “core software” mean for a project which is not yet a project

ILC Software: Why now?

Ties Behnke, DESY

Software becomes an integral part of the “detector development”

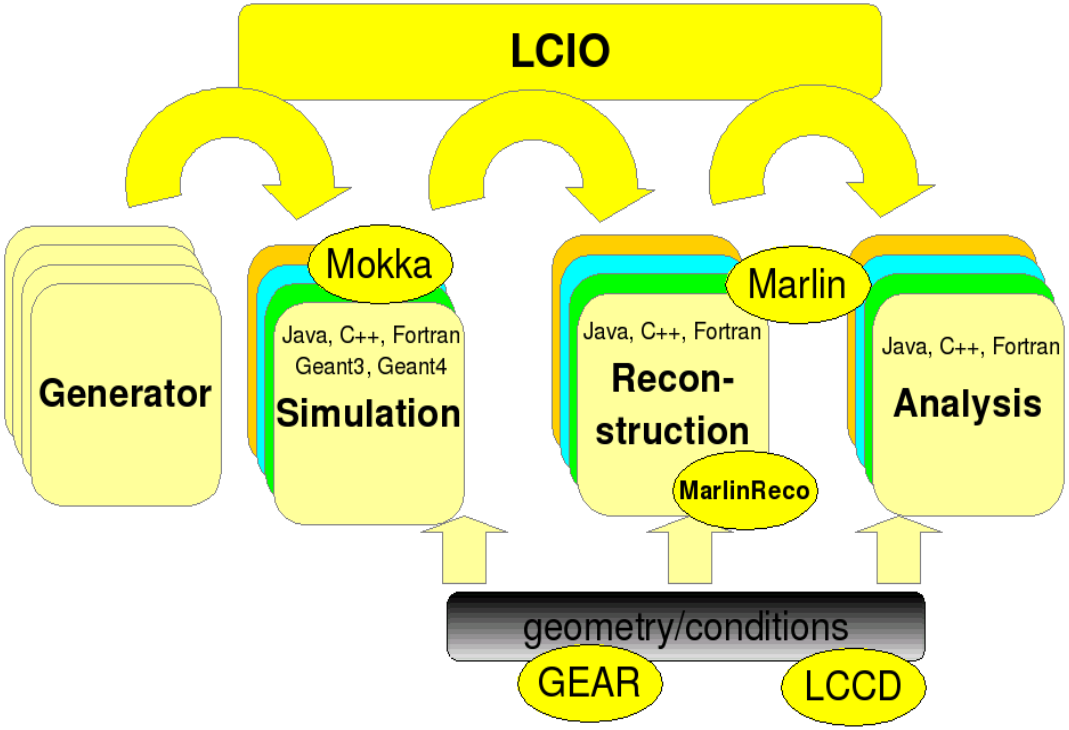
- close relation between optimization and software
- algorithm development becomes centrally important
- data management becomes an important topic early on

Requirements

- easy to use, easy to install, easy to run
- has to be done in “my own” preferred system and language

Where are we?

- Core software packages - status and latest developments
  - LCIO
    - data model & persistency
  - MARLIN
    - C++ application framework
  - LCCD
    - conditions data toolkit
  - GEAR
    - geometry description
  - LCGO
    - a proposal for a new geometry package

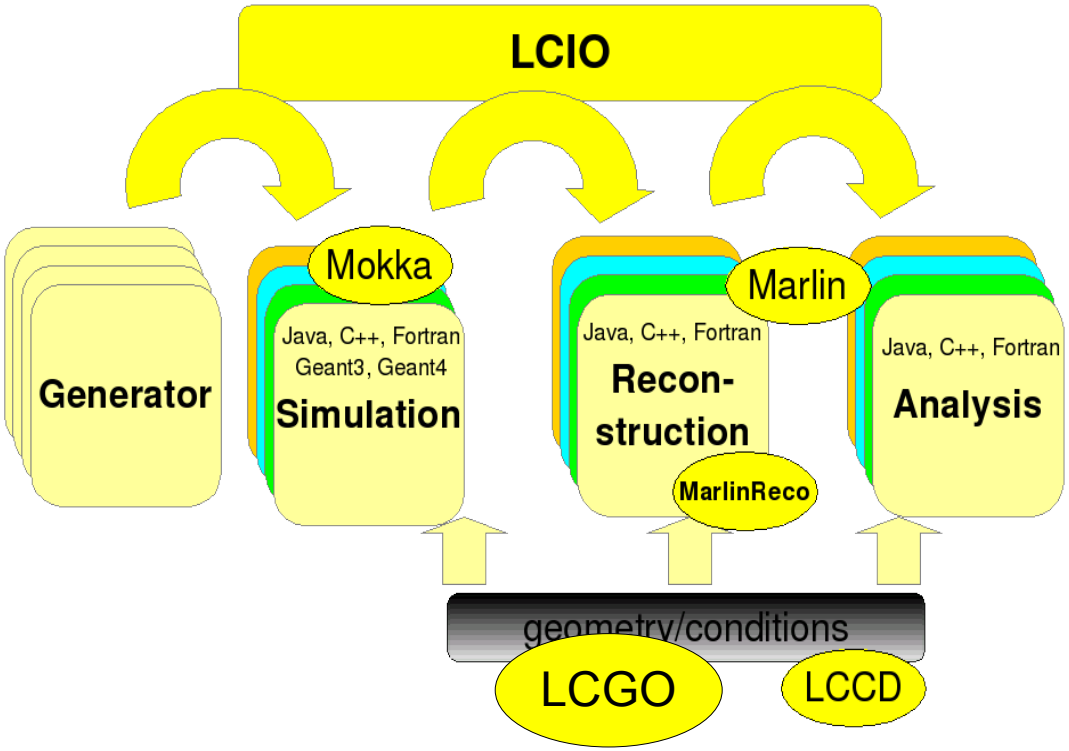


F.Gaede

+ GEANT4, stdhep, clhep, gsl, ... (external packages)

Where are we?

- Core software packages - status and latest developments
  - LCIO
    - data model & persistency
  - MARLIN
    - C++ application framework
  - LCCD
    - conditions data toolkit
  - GEAR
    - geometry description
  - LCGO
    - a proposal for a new geometry package



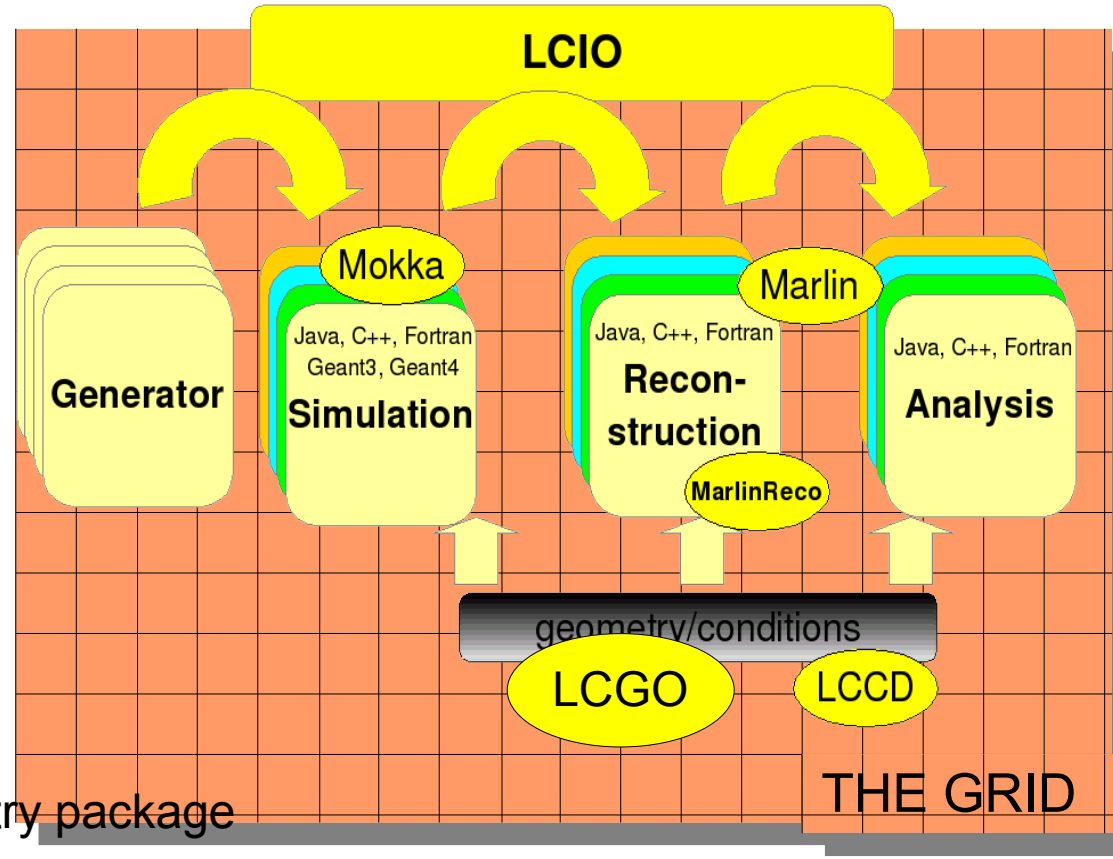
F.Gaede

+ GEANT4, stdhep, clhep, gsl, ... (external packages)

Where are we?

- Core software packages - status and latest developments

  - LCIO
    - data model & persistency
  - MARLIN
    - C++ application framework
  - LCCD
    - conditions data toolkit
  - GEAR
    - geometry description
  - LCGO
    - a proposal for a new geometry package



F.Gaede

+ GEANT4, stdhep, clhep, gsl, ... (external packages)

The basic concept

Based on C++ as core language (widely accepted, though not easy, large common base with LHC)

Lightweight tools, well defined functionality, no large overhead

re-use existing work as much as possible

leave maximum space for personal freedom in development

follow very much the toolbox approach (as opposed to the complete package approach)

Integrate with the GRID for data management and processing

Core packages provide the user with a frame
functionality needs to be filled in by the community / user

Interlude: Interfaces

The definition of interfaces is (one of) the main task(s) of core software:

Examples: LCIO is primarily an interface
AIDA is an histogramming interface
GEAR is a geometry interface
LCGO will be a geometry interface
.....

- Independent from the implementation (SIO in LCIO, ROOT in AIDA, ...)
- Software remains portable and adaptable
- Scales with the number of systems and complexities
- Allows cooperative developments
- Might eventually allow multi-language support

Practical implications

Interfaces need to be defined:

Significant amount of work
usually defined interfaces do not answer your immediate needs...

Interfaces have to be accepted by the developers and users

Saving are not immediatly apparent
often it is seen as restrictive and slowing down the work

“I need to get this information from processor A to processor B, therefore I created a static class .....”
“I cannot be bothered with dealing with the interface, is slows down my work...”

Practical implications

Interfaces need to be defined:

Significant amount of work
usually defined interfaces do not answer your immediate needs...

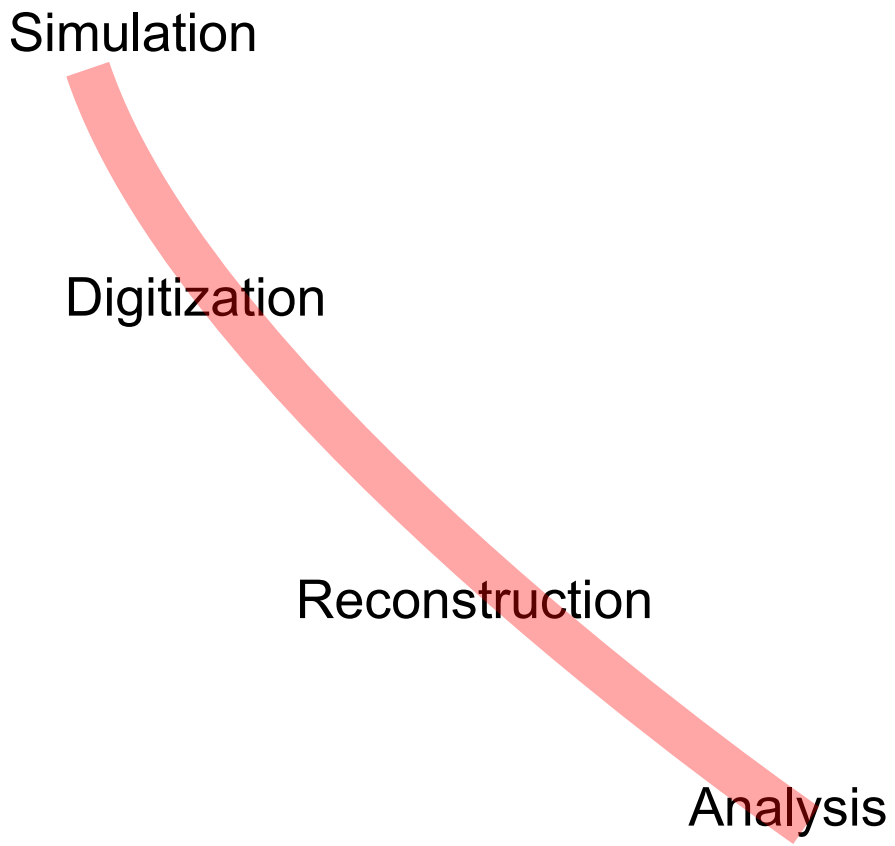
Interfaces have to be accepted by the developers and users

Saving are not immediatly apparent
often it is seen as restrictive and slowing down the work

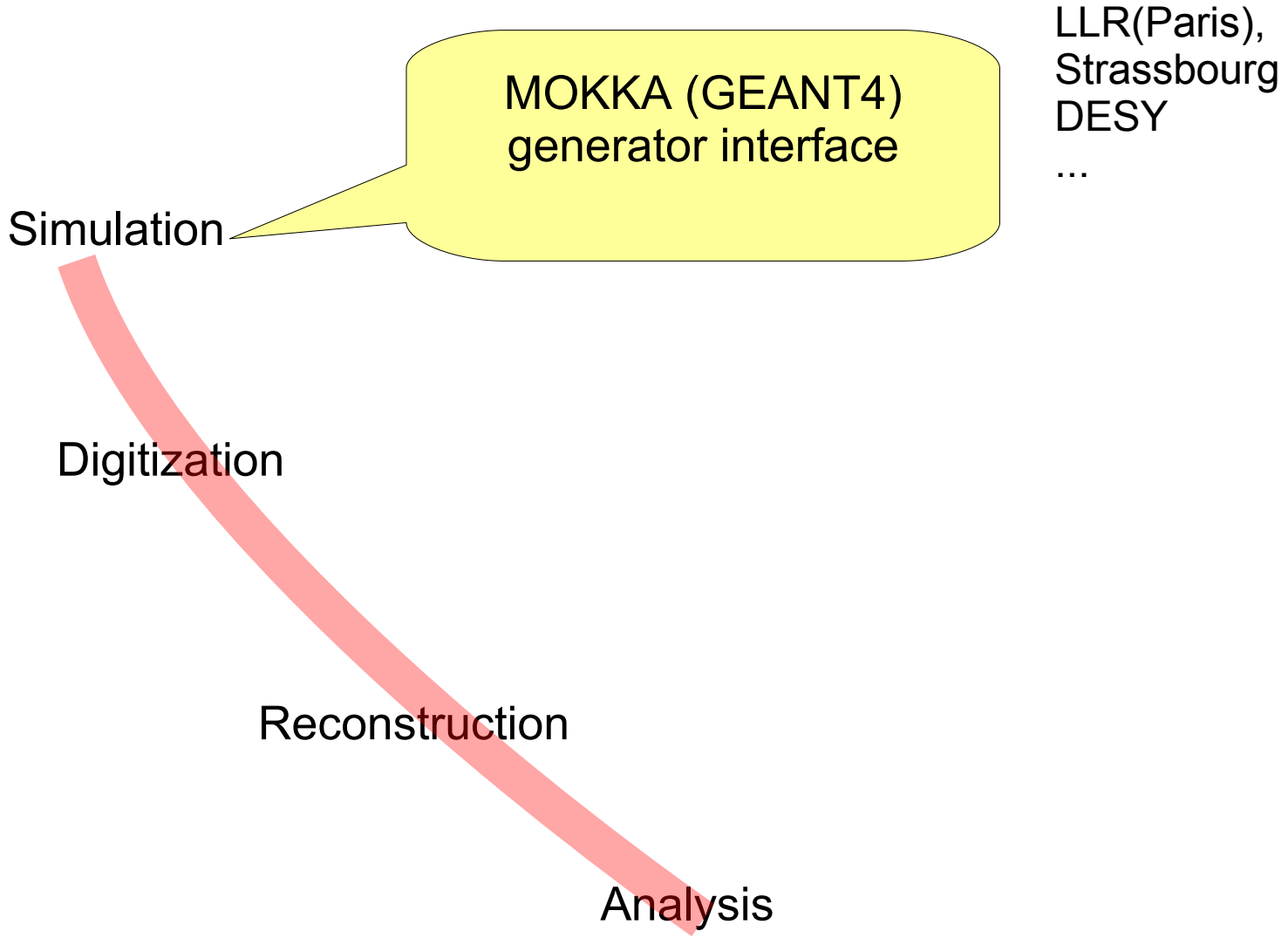
“I need to get this information from [illegible]ssor A to processor B, therefore I created a static cl[illegible]
“I cannot be bothered wit[illegible]g with the interface, is slows down my work...”

Its ALL or NOTHING!!

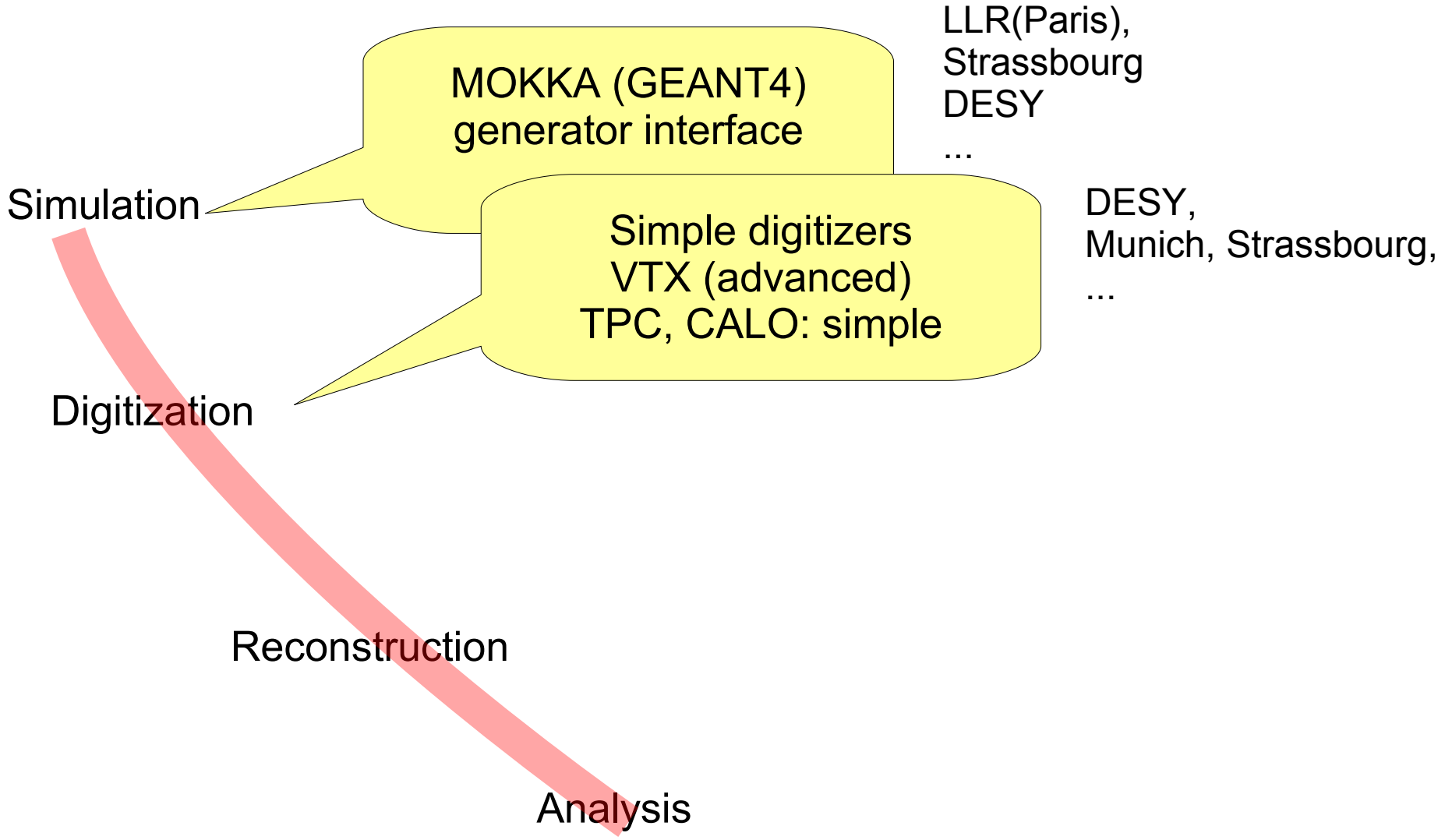
What is there?



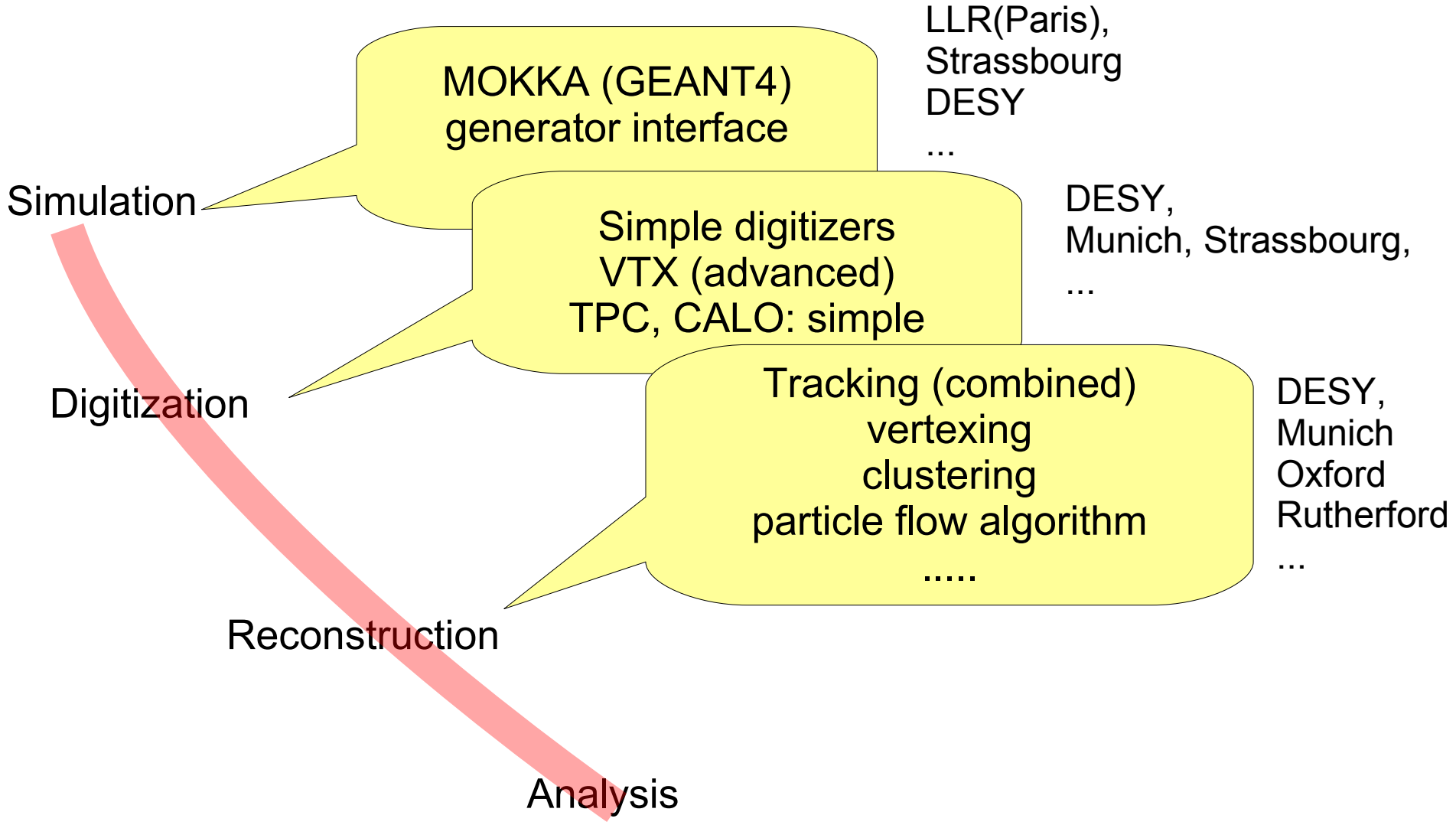
What is there?



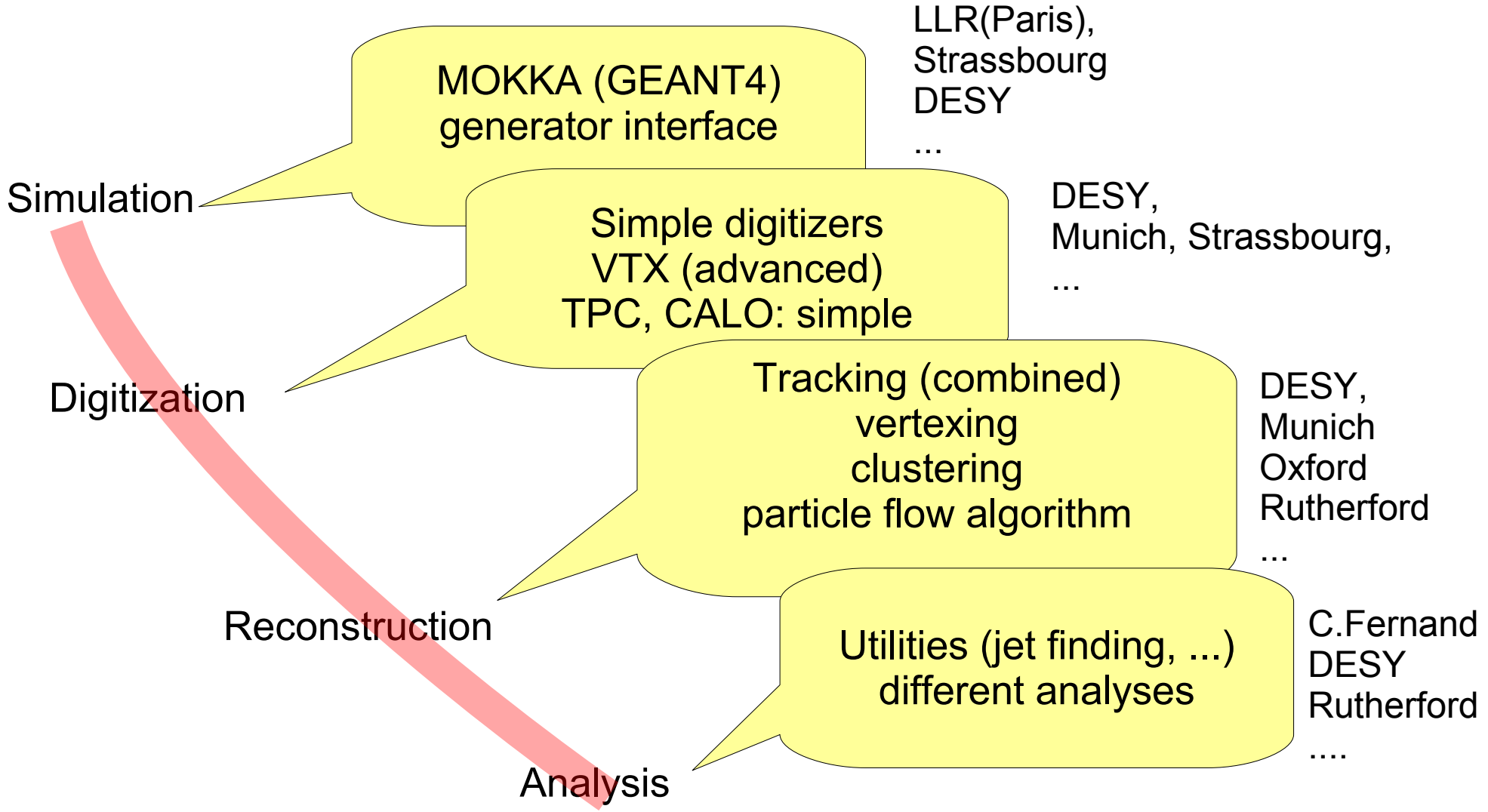
What is there?



What is there?



What is there?



MARLIN-RECO

Suite of reconstruction tools
based on the MARLIN framework

In this sense MARLIN-Reco is one implementation of reconstruction code within a given software model

Provides in one package

- Tracking
- Vertexing
- Calorimeter reconstruction
- Particle flow
- Helper applications

Fairly complete,

modularity allows the user to built its own if he / she so wants to do that

What is not there

Main problem: coherent and easy to use geometry interface

LCGO is designed to close this gap, but it is not yet there
(common DESY/ SLAC development)

State of Algorithms:

- Many are there, but need optimization, debugging, and your ideas
- This is particularly true for particle flow:

  WOLF, PandoraPFA, track based PFA
  many systems, but do not expect something which is “ready”

- Many useful tools are missing

Running the Software

Supported systems:

primary system at the moment is linux

The software has been shown to run under windows, but this is not actively supported (lack of resources)

How to install and run the software:

use the software portal http://ilcsoft.desy.de

In the future: if you have access to afs, use centrally installed afs software under linux and download / install/ link against afs

Some examples

At this workshop:

did see quite a number of analyses which have been started
and which are based on the different pieces of software available

Examples:

several Higgs analyses (Higgs recoil, hadronic Higgs decays,
Higgs self coupling ...)
Our old-time favorite: WW/ ZZ separation in the hadronic channel

and many more technical studies:
photon finding
pi0 finding
vertexing (b/c tagging, VTX charge)

and probably many more:

If you are working on an analysis, please let me know! We like to get an overview of the studies ongoing at the moment

Alternatives

Other ILC software systems available:

- JAVA based system (mostly US, SLAC)
  contact Norman Graf here at the conference
- Root based software systems

  (KEK (GLD) system, 4th concept approach)

A remark: systems which are based on LCIO / interfaced to LCIO allow the exchange and collaboration also in the area of software and algorithm development

Alternatives

Other ILC software systems available:

- JAVA based system (mostly US, SLAC)
  contact Norman Graf here at the conference

LCIO

- Root based software systems

  (KEK (GLD) system, 4th concept approach)

A remark: systems which are based on LCIO / interfaced to LCIO allow the exchange and collaboration also in the area of software and algorithm development

Conclusion

- A coherent software Ansatz exists for ILC studies
- It is fairly lightweight, fairly easy to use
- It is flexible and scalable
- It can be used (though it is far from perfect)
  - But it can only improve if it is used
  - And if everyone contributes to the “tool box” with new tools and gadgets

We should stop to develop only for ourselves,
and we improve the methods to develop towards a common goal:

demonstrate the great potential of the ILC and the ILC detector!