Review Follow-up

Status and plans for implementing the recommendations

Niels Meyer, DESY

CALICE Meeting, Manchester

September 9, 2008

Color code: Recommendation Status *personal opinion NM*

Personell

Review committe expresses need for extra personell officially assigned:

- One person per detector responsible for reconstruction code and conditions data, including testing, documentation, and version control
- At least two more people capable of running the full processing chain
- Database clean-up task force (temporary)

People actually available are very limited:

Simulation	Gabriel Musat
Tracking	Paul Dauncey
SiW ECal	Roman Pöschl
AHCal	Angela Lucaci
TCMT	N.N.
SciECal, DHCals	N.N.

I see this as critical, this is not even enough for maintenance, not talking about further developments. Situation is even worse considering that tracking and SiW ECal only have contact persons, but no team behind.

Self-organization

Some advice for more efficient organization given:

- Name responsibles for individual detectors
  - ✓ See list on slide 'Personell'
- Dedicated software meetings
  - ✓ Started ~2 months ago
- Better documentation and information flow
  - ✓ Central wiki web-page at CERN (thanks to Nigel Watson)
  - ✗ Documentation of existing code as poor as before
- Clarify responsibilities between software and physics coordinators
  - ✗ We of course talk to each other, but nothing official happened, yet.
- Define and schedule analysis goals
  - ✗ Nothing happened, yet
- Closer collaboration with ILC core software
  - ✓ Regular exchange with Frank Gäde, plans to integrate into ILCSoft

Recommended Changes

Objective data quality evaluation and run selection

Analysis started, storage and access still missing

Steering files for reconstruction are long and complicated

Reco files should be processable w/o database access

Common geometry information for reco and simulation *(+analysis)*

User-friendly access to conditions information

Steering Files and Database

There are two concurring user requests frequently expressed:

1. simpler steering files and automatic loading of conditions data
2. processing data w/o database access

To some extent, this results from mis-understanding the ideas of LCCD, the interface used to include conditions into event data

Conditions data is stored in LCCollections (like event data), and gets loaded at run time. All code using this data is transparent to the physical origin of these collections, defined in the steering

Alternative: develop tool to load conditions data automatically, which realistically loads all conditions data (overkill?) from the database

There is NO WAY to fullfill request 1. and 2. at the same time !!!

Complicated Steering Files

New tool: set of shell scripts to generate steering files for reconstruction from minimum amount of information

available cvs: `cvs co calice_tools/calice_run`

example call:

```
   generateRecoSteering 500123 local
```

features:

- Includes all files ./Run500123.???.slcio as input files
- Launches ECal, HCal, and TCMT reco depending on run-type
- Loads all necessary conditions data from correct db folders

planned:

- Extend to allow arbitrary CALICE jobs (requires user templates)
- Software installation, run script, job submission to GRID

DB-Independent Reco Files

LCCD allows to store conditions data together with event data, so technically it is possible to ship conditions data with reco files

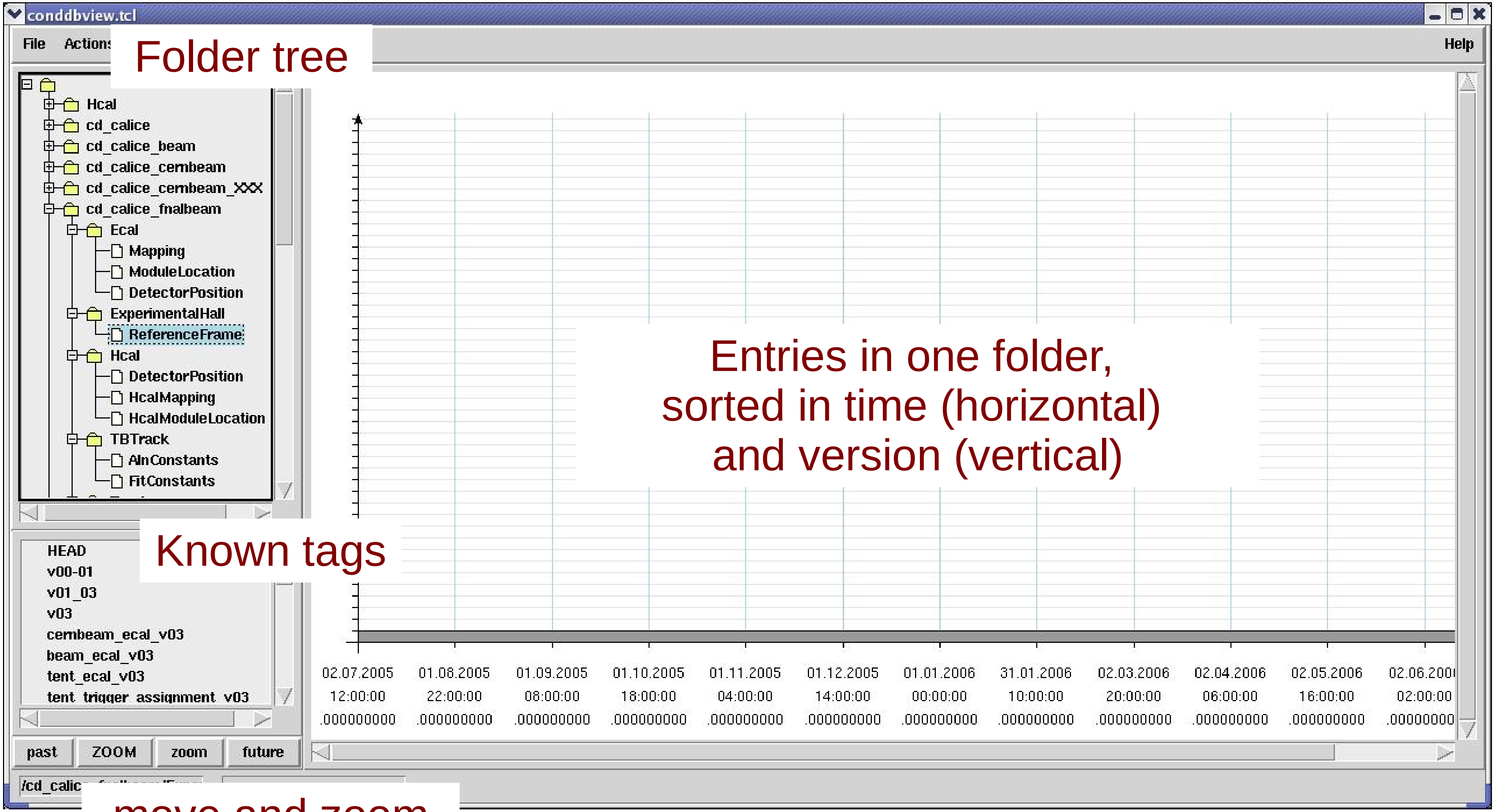
Discussed this in software meeting

Seems feasible for reco files, not for raw slcio files. Not exactly clear how much effort will be necessary, feel free to join!

Danger is: conditions data in reco files may be obsolete. Always possible to superseed with latest status from database, but this needs understanding how to modify the steering files

The database will still be integral part of our data handling!

Database viewer



```
⊟ 🗀
  ⊞ 🗀 Hcal
  ⊞ 🗀 cd_calice
  ⊞ 🗀 cd_calice_beam
  ⊞ 🗀 cd_calice_cernbeam
  ⊞ 🗀 cd_calice_cernbeam_XXX
  ⊟ 🗀 cd_calice_fnalbeam
     ⊟ 🗀 Ecal
          🗋 Mapping
          🗋 ModuleLocation
          🗋 DetectorPosition
     ⊟ 🗀 ExperimentalHall
          🗋 ReferenceFrame
     ⊟ 🗀 Hcal
          🗋 DetectorPosition
          🗋 HcalMapping
          🗋 HcalModuleLocation
     ⊟ 🗀 TBTrack
          🗋 AlnConstants
          🗋 FitConstants
```

```
HEAD
v00-01
v01_03
v03
cernbeam_ecal_v03
beam_ecal_v03
tent_ecal_v03
tent_trigger_assignment_v03
```

/cd_calic

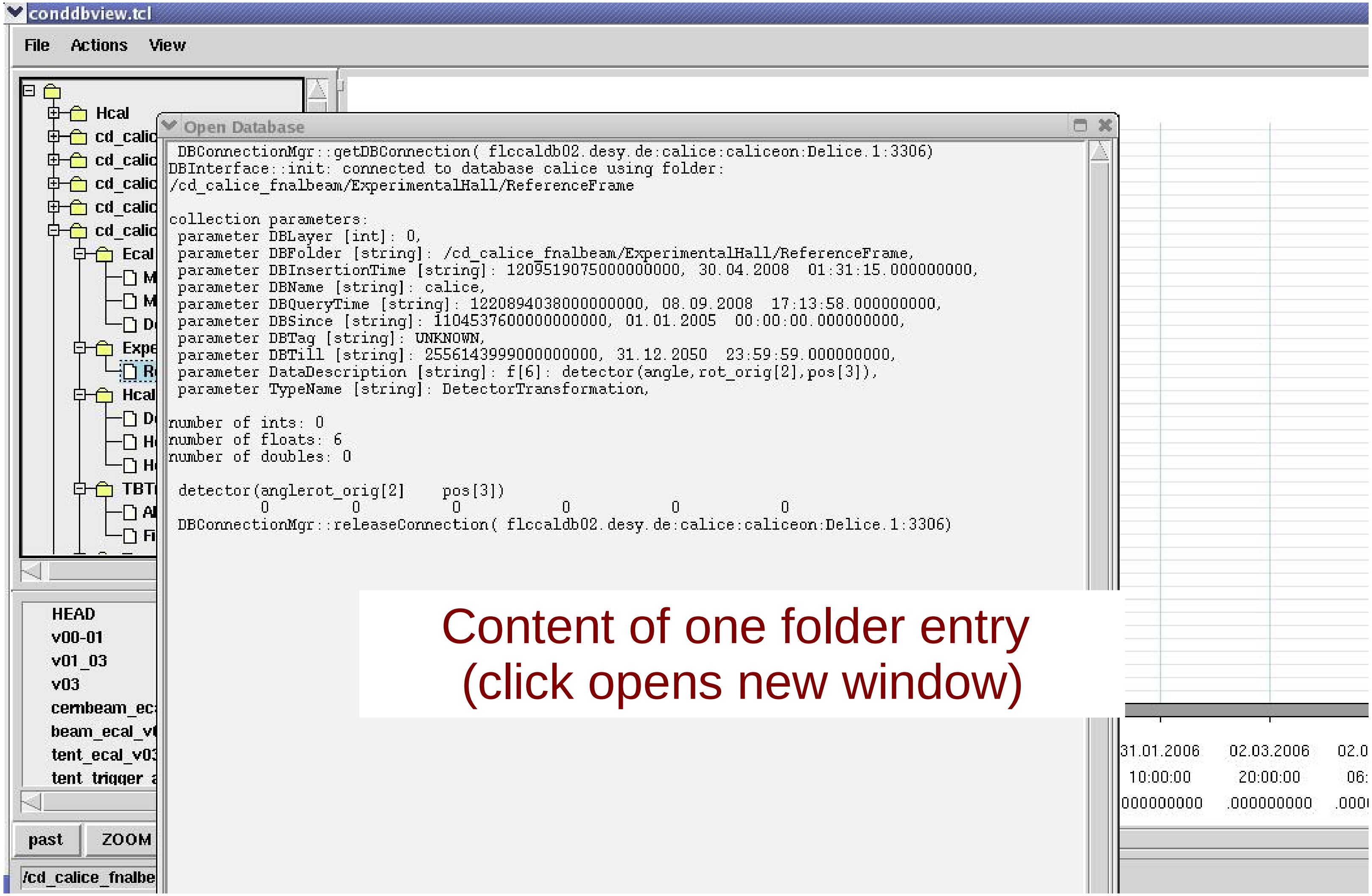
Folder tree

Known tags

Entries in one folder, sorted in time (horizontal) and version (vertical)

move and zoom
for 'time' axis

Database viewer

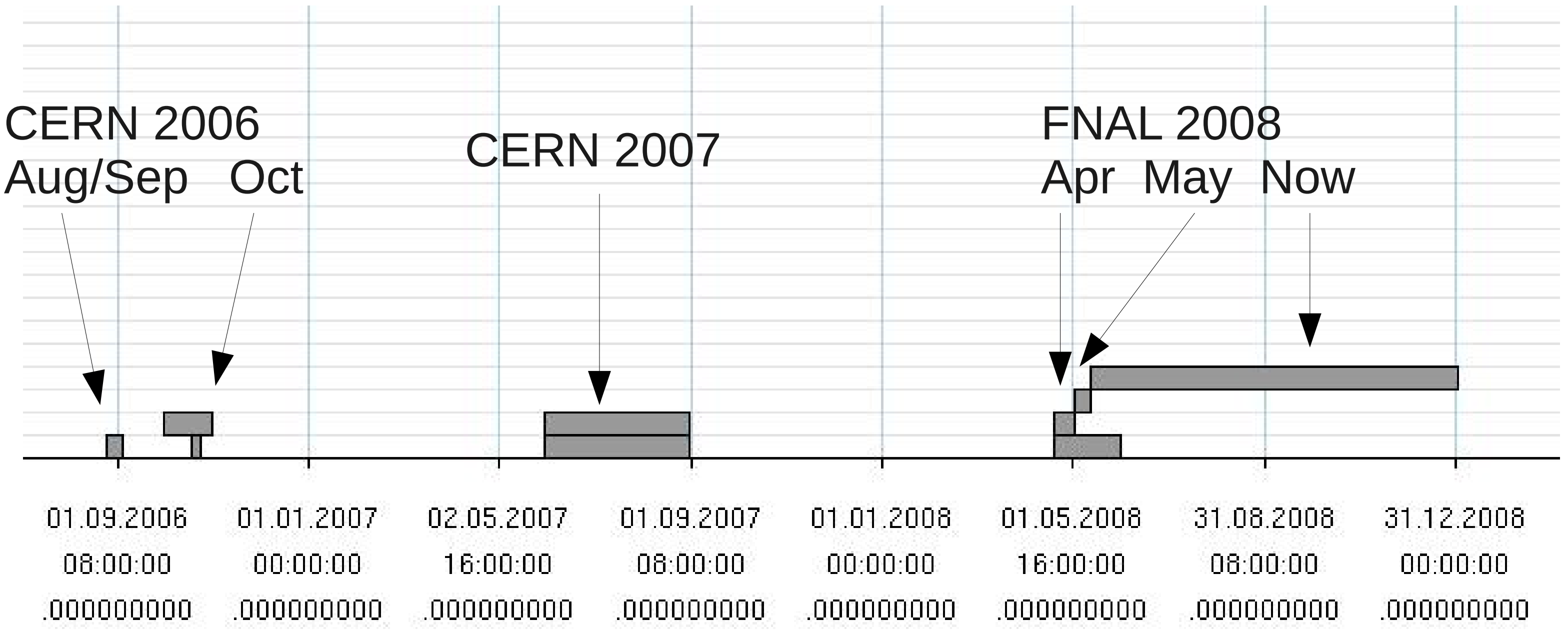


```
 DBConnectionMgr::getDBConnection( flccaldb02.desy.de:calice:caliceon:Delice.1:3306)
DBInterface::init: connected to database calice using folder:
/cd_calice_fnalbeam/ExperimentalHall/ReferenceFrame

collection parameters:
 parameter DBLayer [int]: 0,
 parameter DBFolder [string]: /cd_calice_fnalbeam/ExperimentalHall/ReferenceFrame,
 parameter DBInsertionTime [string]: 1209519075000000000, 30.04.2008  01:31:15.000000000,
 parameter DBName [string]: calice,
 parameter DBQueryTime [string]: 1220894038000000000, 08.09.2008  17:13:58.000000000,
 parameter DBSince [string]: 1104537600000000000, 01.01.2005  00:00:00.000000000,
 parameter DBTag [string]: UNKNOWN,
 parameter DBTill [string]: 2556143999000000000, 31.12.2050  23:59:59.000000000,
 parameter DataDescription [string]: f[6]: detector(angle,rot_orig[2],pos[3]),
 parameter TypeName [string]: DetectorTransformation,

number of ints: 0
number of floats: 6
number of doubles: 0

 detector(anglerot_orig[2]    pos[3])
          0         0          0           0           0           0
 DBConnectionMgr::releaseConnection( flccaldb02.desy.de:calice:caliceon:Delice.1:3306)
```

Content of one folder entry
(click opens new window)

Database viewer

Example: HCal gain calibrations for one half module
/cd_calice/Hcal/Gain/Approved/ID09A



Used by experts already for long, but now available from CVS:

```
$> cvs co calice_tools/dbview
```

Geometry

A long list of recommendation is directly or indirectly connected to a transparant and flexible geometry description of our setup:

- Common source for geometry parameters in reco and simulation
- Coherent channel numbering
- Missing event display
- Missing high level analysis
- Feedback to ILD optimization
- User-friendly access to conditions information

Channel Numbering

Recommendation: coherent cell indexing

Current geometrical cell index (written to output hits) encodes
layer, wafer row, wafer column, pad row, pad column

This is more a 'hardware cell index' for the SiW ECal. Proposal:

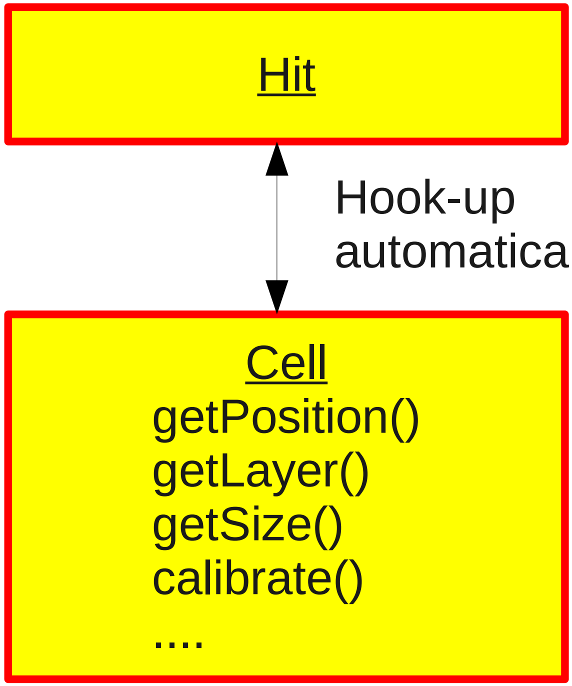
Geometrical index:
Encodes detector, layer, horizontal position, vertical position and gets used for data/MC matching ⇒ the index of hits

Hardware index:
Can be chosen freely by each detector, identifier for calibration constants and similar device properties

In principle also needed: common DAQ index for input channel

Translation from one index to another has to be transparent!

Ideas...

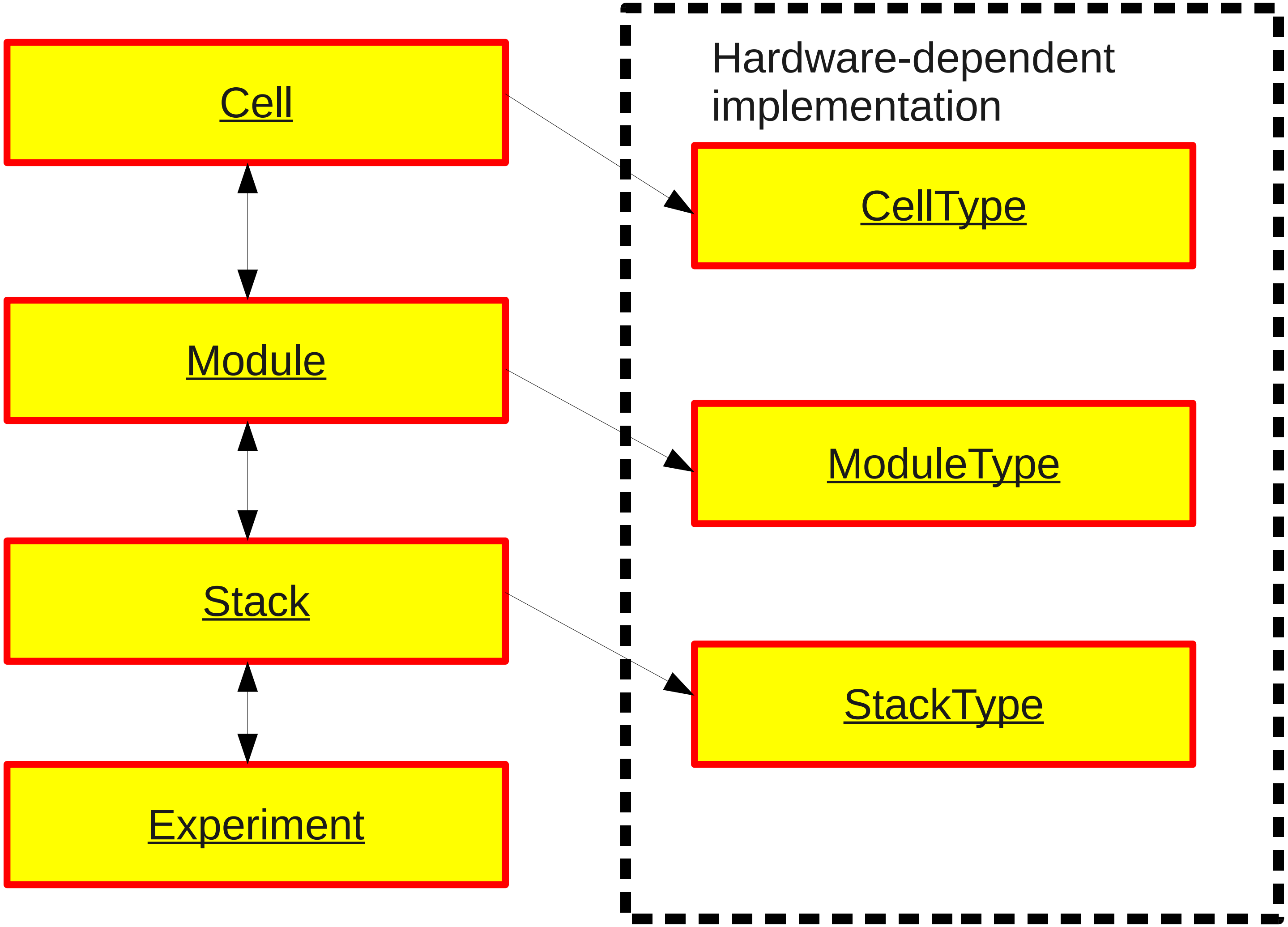


Usually, users start with hits

Amplitude is event data, everything else is 'constant' and belongs to the hardware cell

Some hierarchical structure below the cell seems sensible

Ideas...



Cell

Module

Stack

Experiment

Hardware-dependent implementation

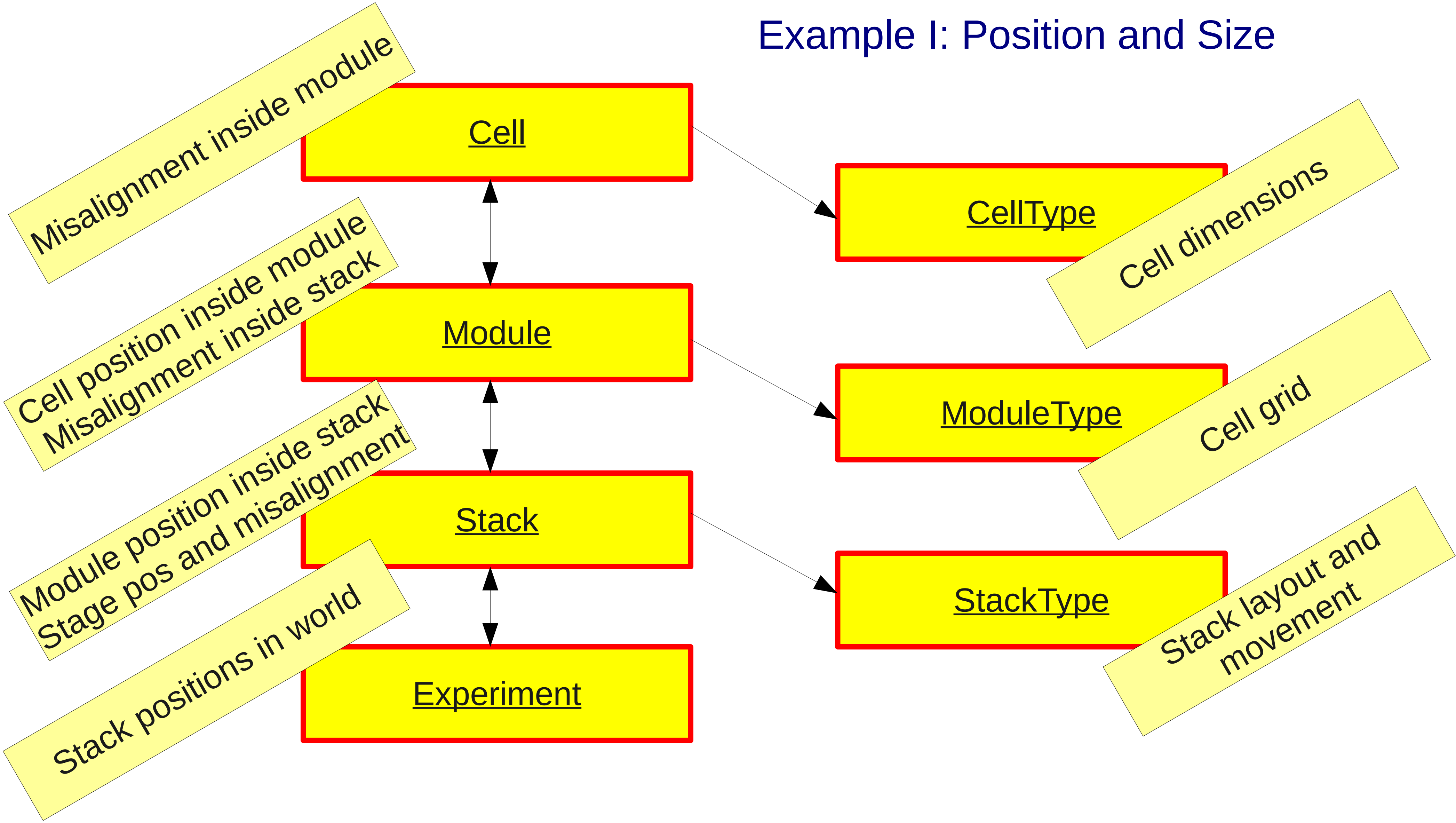
CellType

ModuleType

StackType

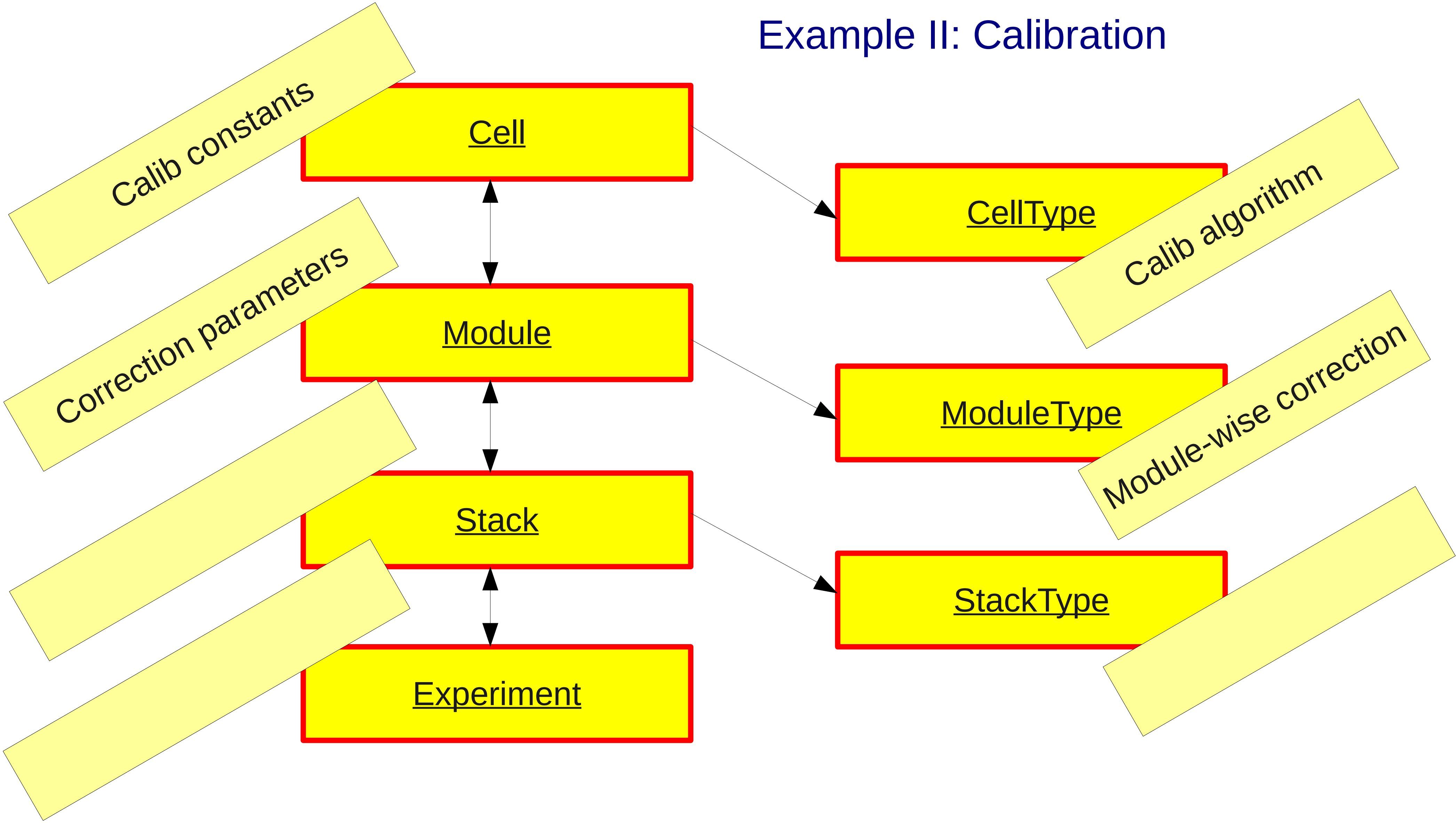
Ideas...

Example I: Position and Size


Misalignment inside module
Cell
CellType
Cell dimensions
Cell position inside module
Misalignment inside stack
Module
ModuleType
Cell grid
Module position inside stack
Stage pos and misalignment
Stack
StackType
Stack layout and movement
Stack positions in world
Experiment

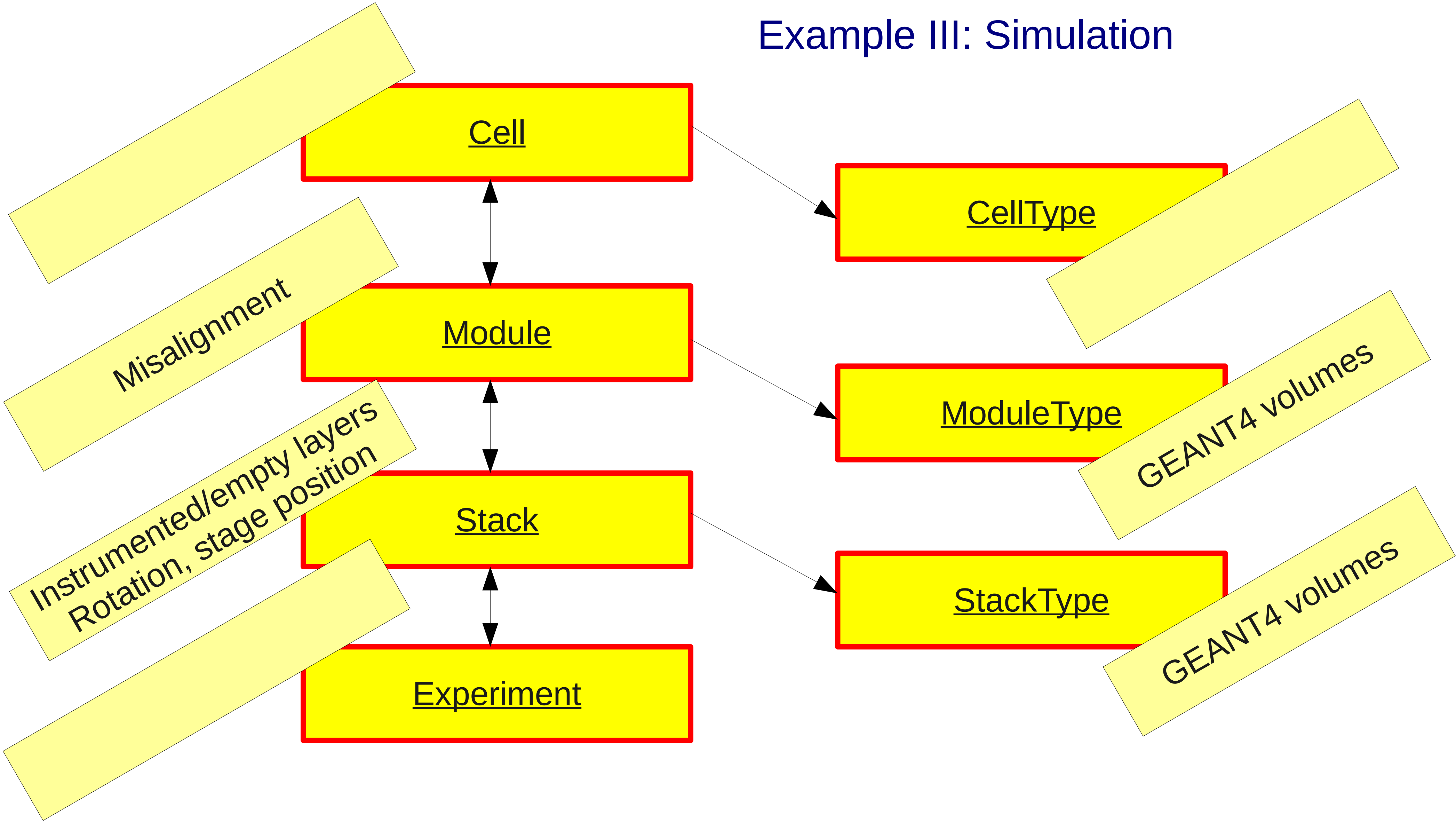
Ideas...

Example II: Calibration



Ideas...

Example III: Simulation



Concluding Remarks

Contact with F. Gäde about integrating the interfaces into GEAR. This seems feasible from the technical point of view

We are interested in fast progress; my recommendation would be to integrate pure geometry into GEAR and to keep the extension bejond 'private' to CALICE

Have to encourage analyzers to implement their tools in re-usable fashion. The hope is that the proposed scheme is of help here

In general, we as a collaboration have to cooperate much more on the software. I see good tendencies (run quality tools), but we need to get much better