# ucesb status

Bastian Löher
March 2017

TU Darmstadt

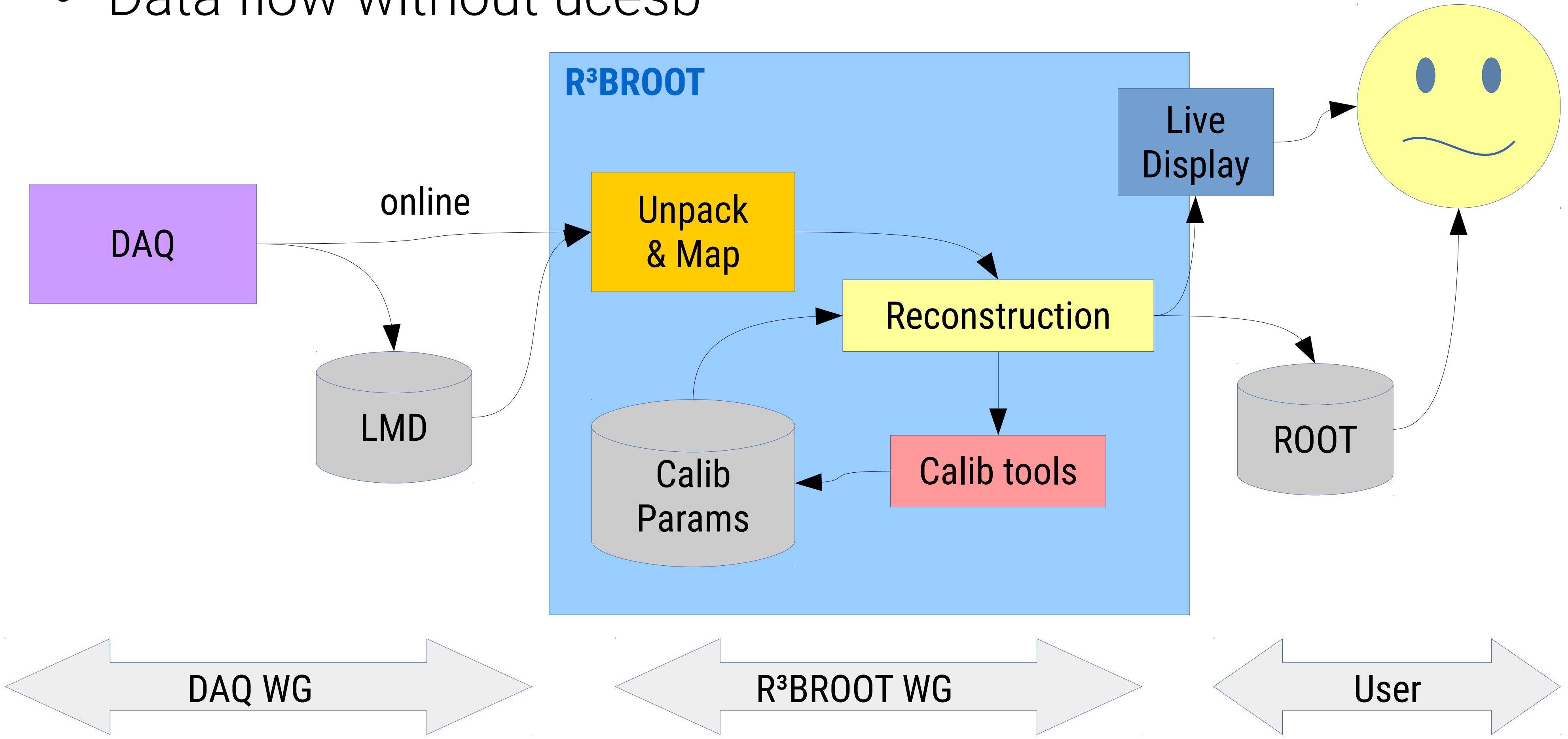
GSI Helmholtzzentrum für Schwerionenforschung GmbH

## Last workshop (July 2015)

- Unpackers for each detector written by hand
- Channel mapping (UNPACK→MAPPED/RAW) implemented individually
- Maybe remember `R3BTofUnpack` class

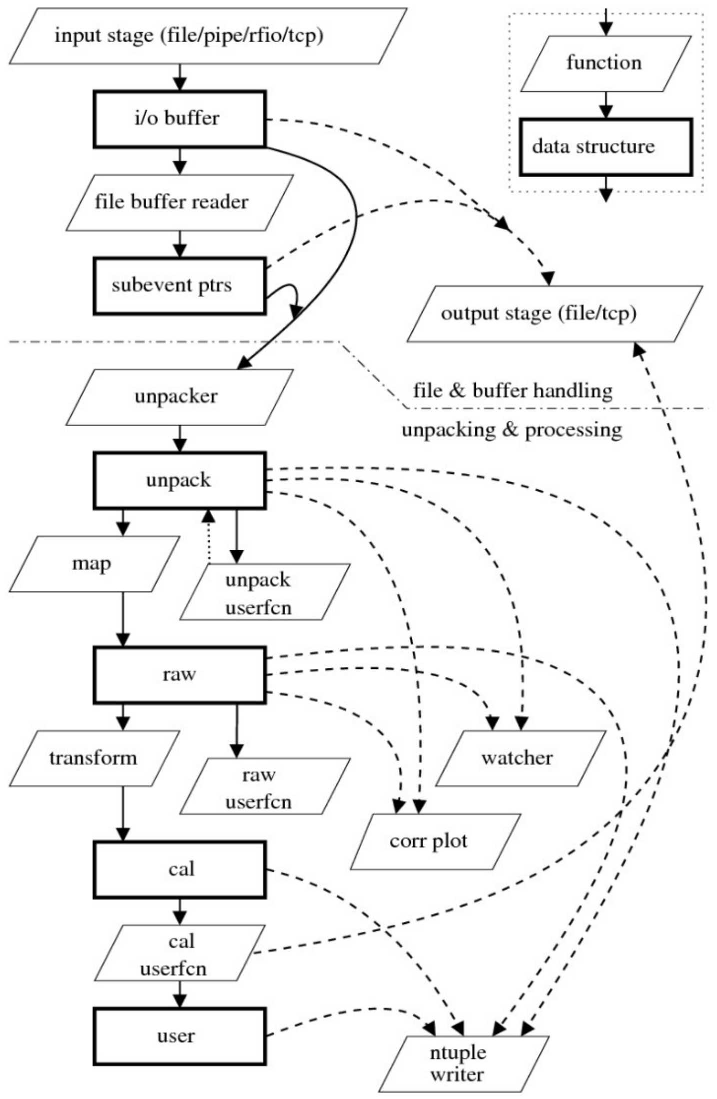
# Last workshop (July 2015)

- Data flow without ucesb



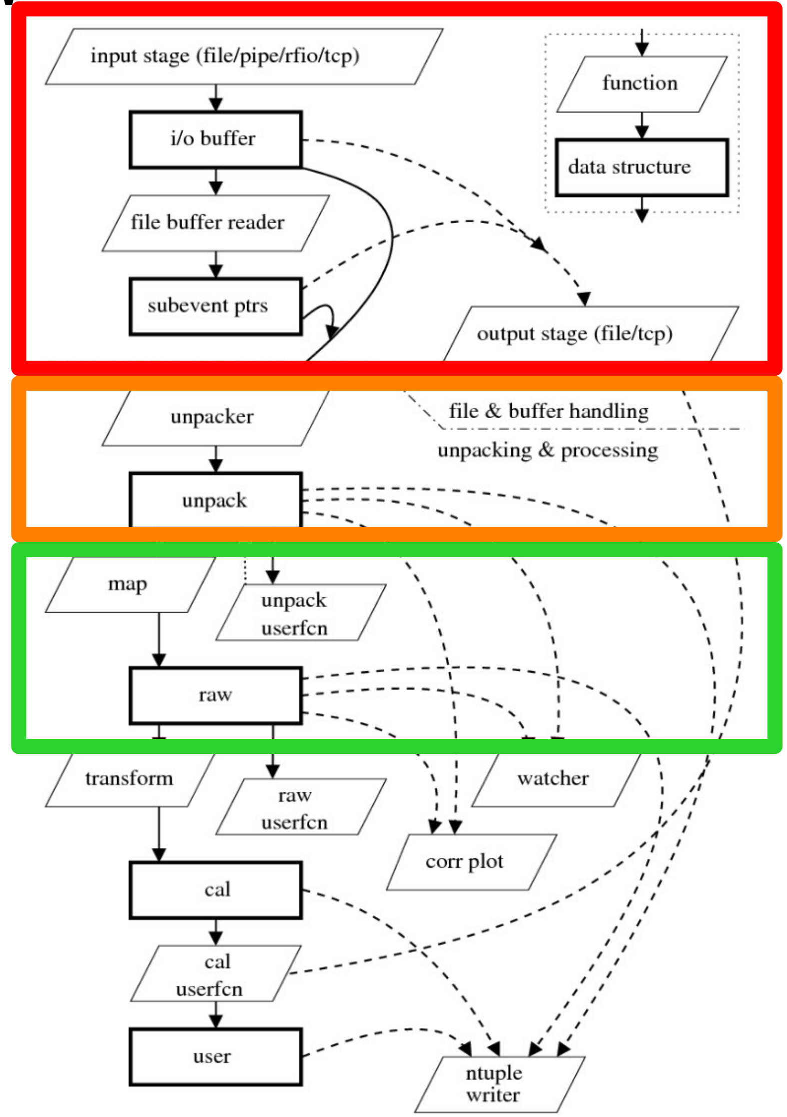
# ucesb overview

- Internal data flow



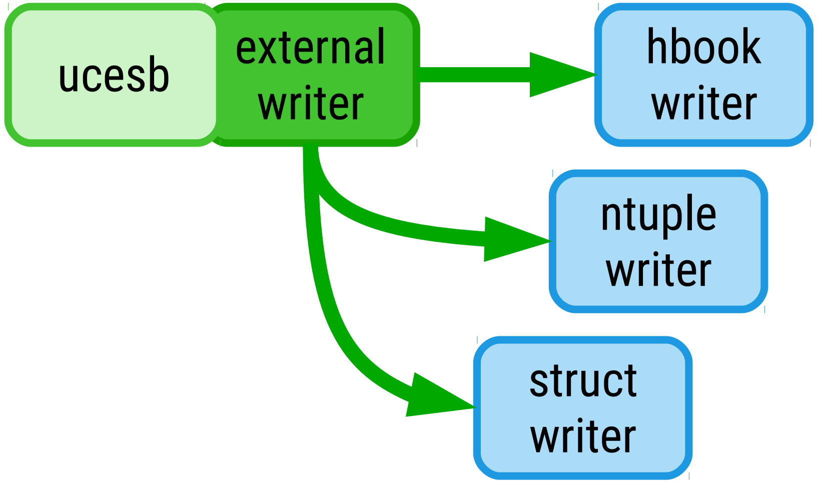
# ucesb overview

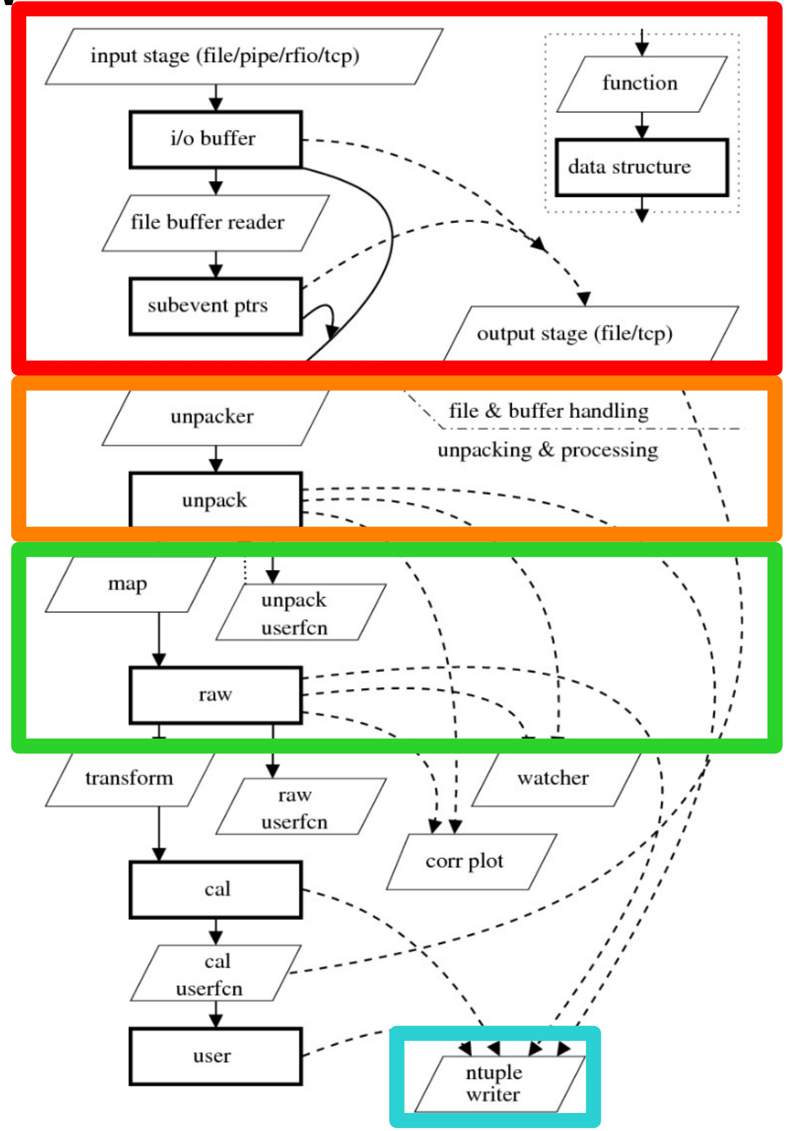
- Internal data flow
- Use input, unpacking and mapping stages



# ucesb overview

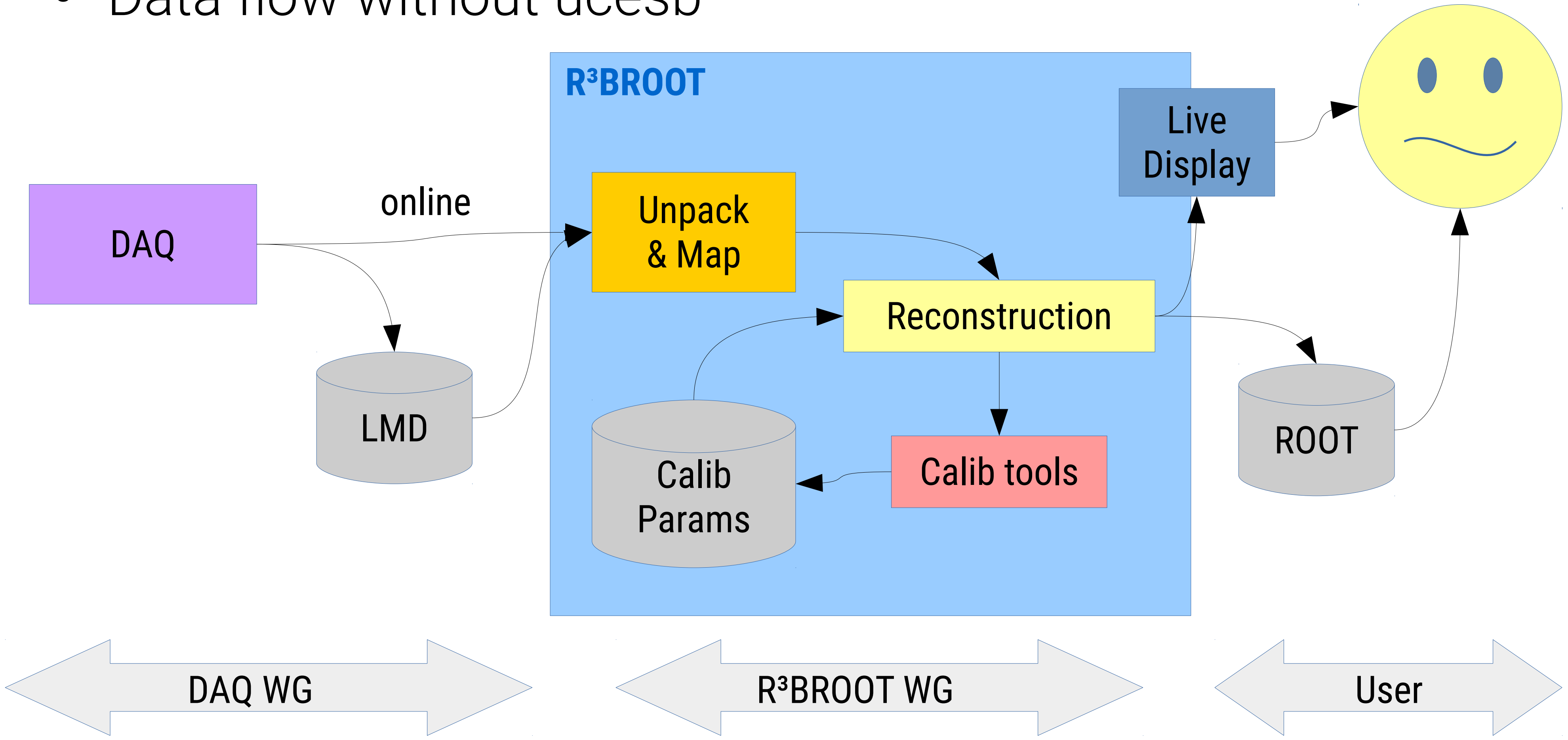
- Internal data flow
- Use input, unpacking and mapping stages
- Write output to file or via network





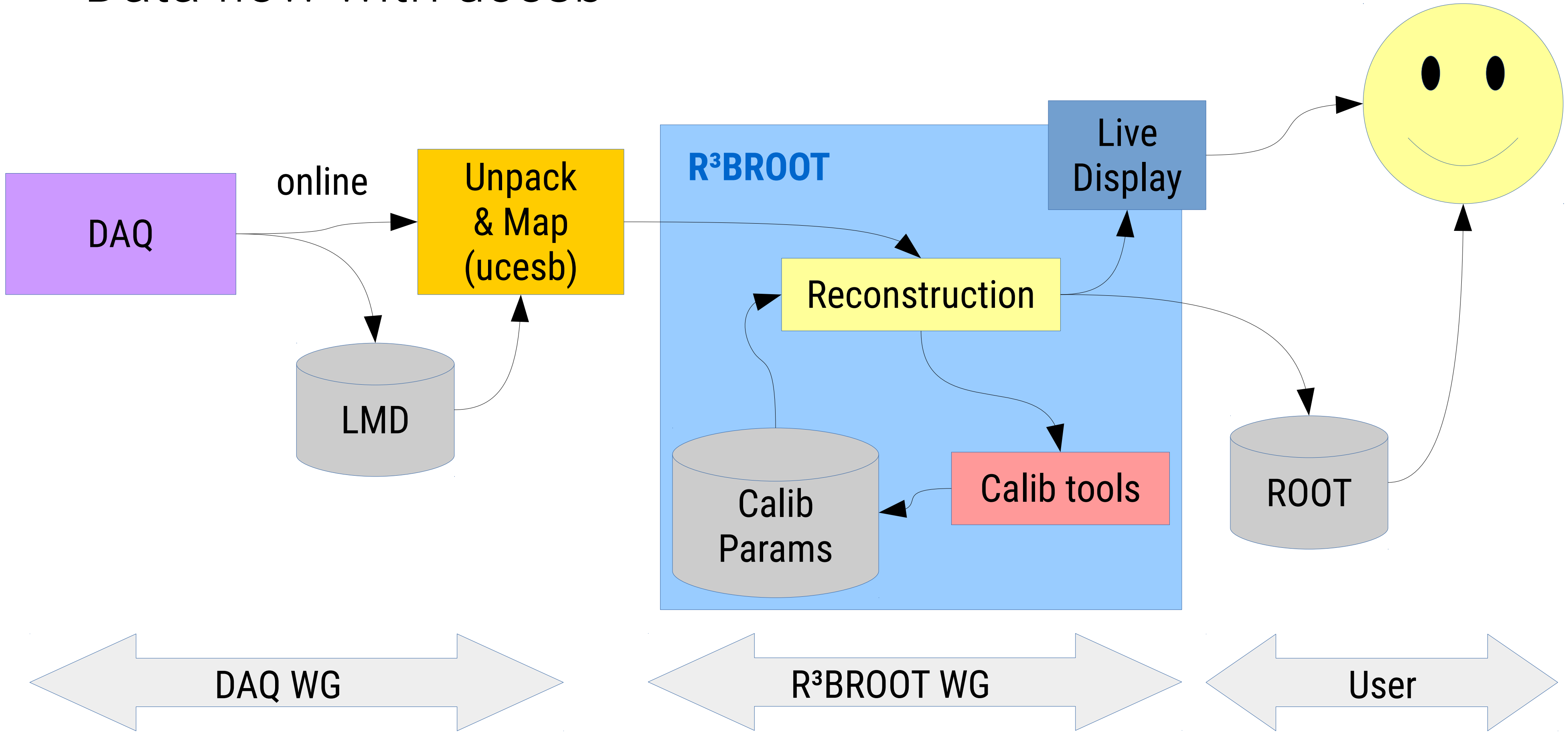
# Last workshop (July 2015)

- Data flow without ucesb



# Today

- Data flow with ucesb

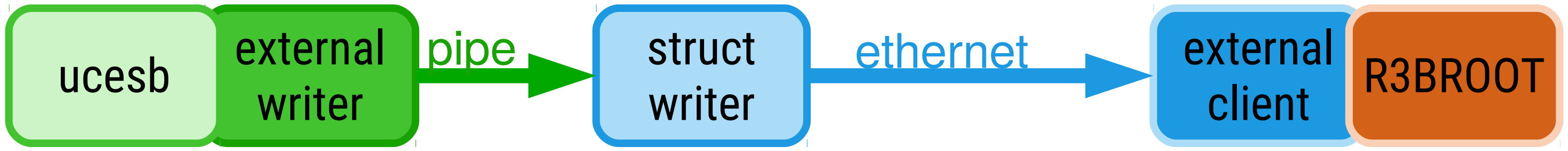


## Advantages

- Unpacker has to be written for proper DAQ operation
  - Why not reuse?
- Mapping has to be defined somewhere
  - Why not do it in a consistent manner?
- Ucesb does extensive checking by default
  - Why reinvent the wheel?
- DAQ WG does most of this work → Less R3BROOT code

# R3BROOT + ucesb

- Data flow



- Implementation (in R3BRoot/r3bsource):
  - R3BUcesbSource: Start unpacker and establish connection via ext_data_client
  - R3BReader: Base class for extracting data from ucesb event
  - R3BPtofReader: Derived detector specific reader class

## The reader class

- Needs data structure (`ext_str_h101_<det>.h`):
  - Generated via ucesb:
    `ucesb –ntuple=DET,STRUCT_HH,ext_str_h101_<det>.h`
  - Needs to be modified (currently by hand), see existing structures in r3bsource directory
- Methods:
  - `Init()`: Calls `EXT_STR_h101_<det>_ITEMS_INFO`
  - `Read()`: Copies data from ucesb structure into R3Broot data containers

## In a Macro

See: https://github.com/R3BRootGroup/R3BRoot/blob/dev/macros/r3b/unpack/ucesb/run_empty.C

- Define full data structure:

```
struct {
    EXT_STR_h101_det1 det1;
    EXT_STR_h101_det2 det2;
} EXT_STR_h101;
```

- Create an instance in `run()` function:

```
EXT_STR_h101 event;
```

## In a Macro

See: https://github.com/R3BRootGroup/R3BRoot/blob/dev/macros/r3b/unpack/ucesb/run_empty.C

- R3BUcesbSource Instantiation:

```
source =
  new R3BUcesbSource(file, options, ucesb, &event, sizeof(event));
```

  - `file`: Input file name (may use wildcards)
  - `options`: additional options to −ntuple option
  - `ucesb`: path to unpacker (use $UCESB_DIR variable)
  - `event`: full event structure

## In a Macro

See: https://github.com/R3BRootGroup/R3BRoot/blob/dev/macros/r3b/unpack/ucesb/run_empty.C

- Reader Instantiation:

```
det2 =
  new Det2Reader(&event.det2, offsetof(EXT_STR_h101, det2));
source→AddReader(det2);
```

  - Adress of detector data structure inside full event → destination address
  - Offset of this address with respect to beginning of structure → source address

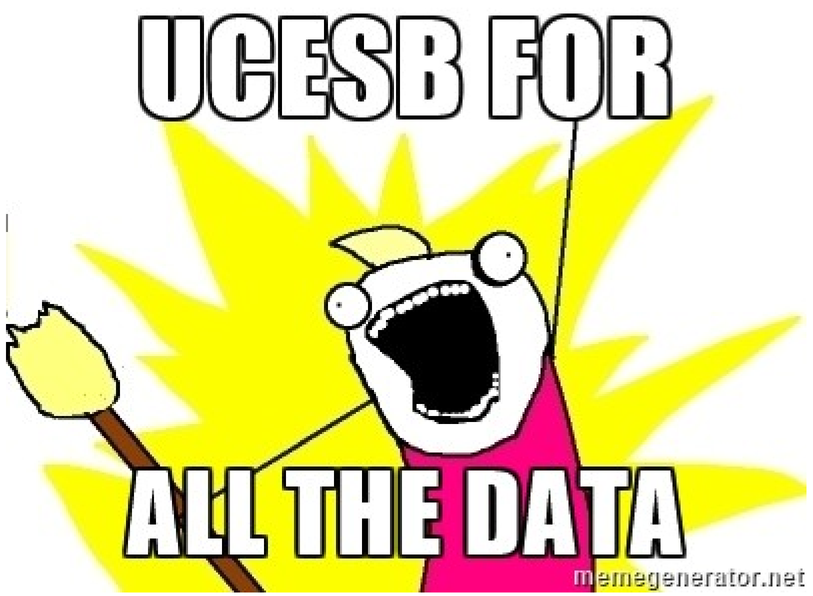
## Which reader classes exist today?

From: https://github.com/R3BRootGroup/R3BRoot/tree/dev/r3bsource

- Fiber4
- Los, rolu
- Neuland (Tacquila & Tamex)
- Psp, Pspx
- Strawtubes
- Tofd
- Unpack (event number + trigger)

## Which detectors could use ucesb?

## Which detectors could use ucesb?



## Example used in this workshop

- Proton ToF detector (Ptof)
- ucesb unpacker: `upexps/jun16/jun16_ptof`
  - Get it here: `git clone /u/land/bloeher/s438b/upexps`
  - Check out: `git checkout for_r3broot_workshop`
- data: jun2016/run160*.lmd (time-stitched, Carbon run)
  - Get it here: `/SAT/nyx/land/jun2016/stitched/…`
  - Or cached: `/d/land2/bloeher/nyx_cache/jun2016/…`
  - Contains data from Tofd detector → similar format for Ptof

## Avoiding Pitfalls

- $UCESB_DIR must be exported when compiling R3BRoot
- ucesb and R3Broot must be compiled with same version of root
- Check that the used ext_str_101_<det>.h matches output of ucesb
- Double check the arguments when instantiating Reader class