# RPC DHCAL 1m$^3$ test software:
## daq, event building, event display, analysis and simulation

Lei Xia

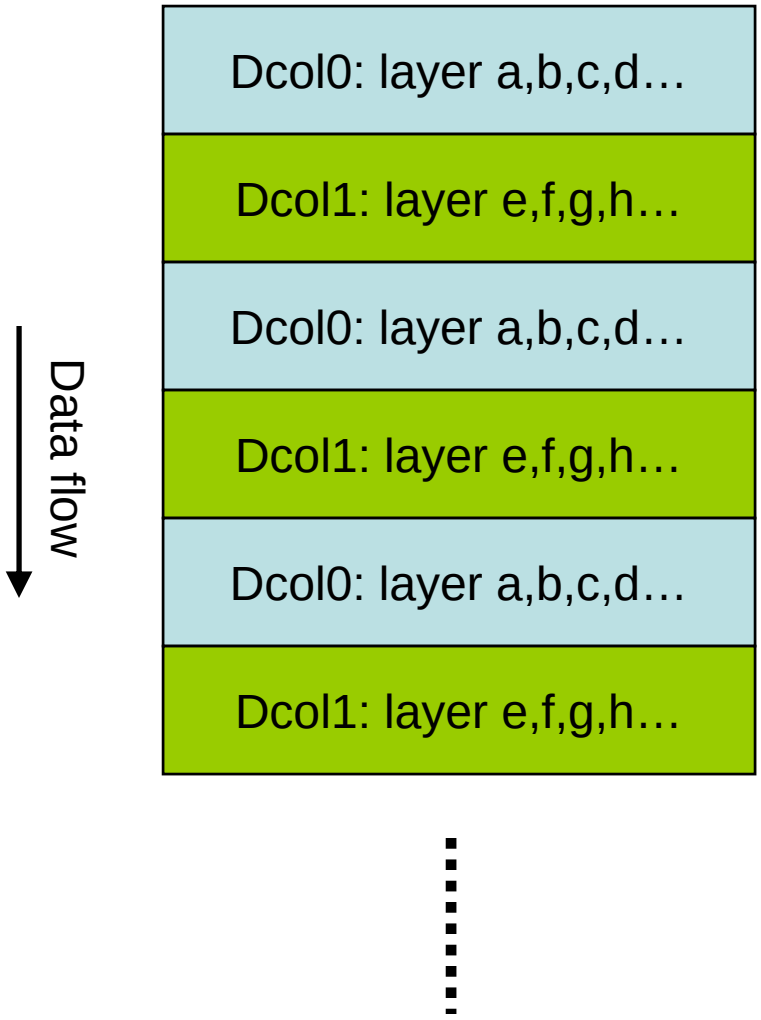
# Daq: status

- Developed a local version of CALICE daq (UK daq) system for RPC slice test, by Jim Schlereth (ANL)
- Has been running with CALICE daq for 1+ years
  - Most tasks went very smoothly
  - Implemented all current run types
  - Implemented automated run submission, parameter scan, etc.
- Some minor issues related to CALICE daq
  - Error handling
    - CALICE daq tries to make runs go smoothly, even in case of hardware error
    - At developing phase, we would rather interrupt runs in some cases
  - Run time calculation
    - CALICE daq count configuration time into run time
    - We would like to exclude that
  - Fix: relatively straight forward
    - Being worked out

# Daq: moving on to 1m$^3$ tests

- One major issue is ‘data record’
  - CALICE daq ‘record’ has a size limit (64KB?)
  - DHCal data comes into daq buffer as a continuous flow
    - No clear boundary between events, nor anything else that naturally divide data into pieces
    - Total data size far exceed current ‘record’ size limit
  - Need a way to chop up the data stream and put each piece into a ‘record’
    - Later, one need to recover the whole stream from the records
    - Without knowing the events, some tasks are not convenient at least (come back to this later)
    - Fast (online) event building? Very risky…
- Other changes needed to 1m$^3$
  - Front End: data format changes (address, register, etc.)
    - Event data:
      - Data bit fields interpreted only by analysis code, some online histogram code
      - Classes affected: DhcFeHitData
    - Configuration data:
      - Container classes representing DCAL registers
      - Classes managing configuration data
  - Trigger: multiple TTM, register change, sync issues, etc
  - Multiple VME crates: supported by CAEN & HAL, need to be tested
  - Daq strategy: fixed readout interval? Or do polling?

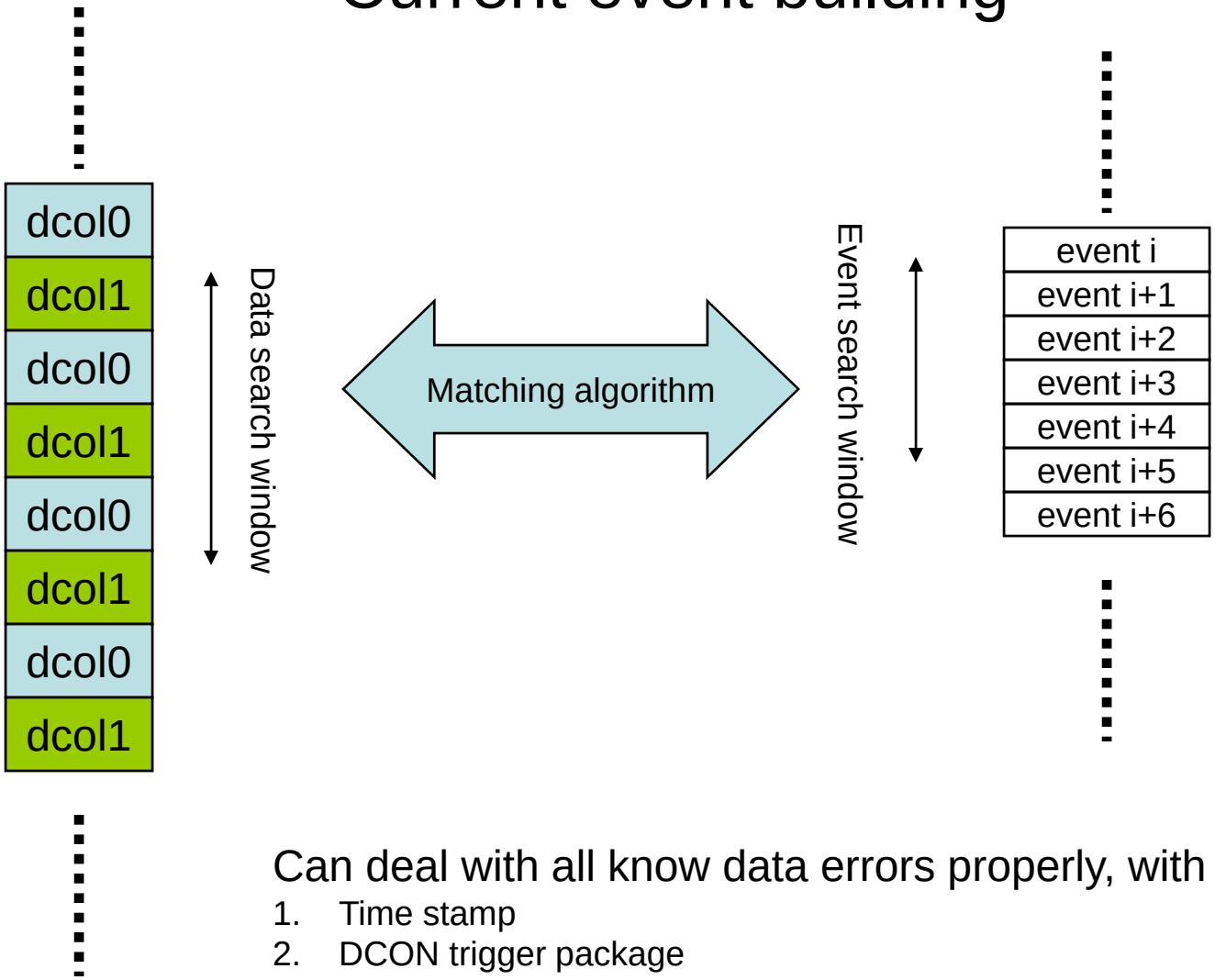
# Event builder: current data structure




N sec worth of data
(for cosmic ray: 0 – n triggers, usually complete)

For each trigger:
each DCON report back a trigger package
each DCAL send back 7 data package
DCON receives data pks, do zero suppression

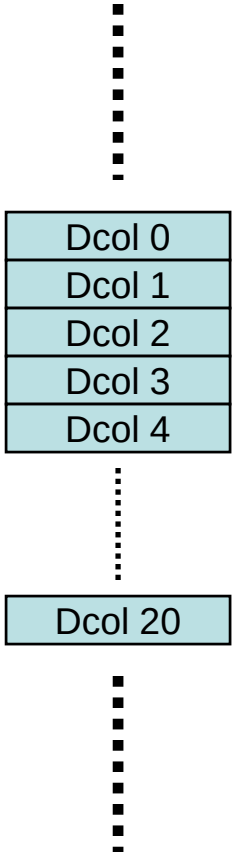
# Current event building




Can deal with all know data errors properly, with the help of:

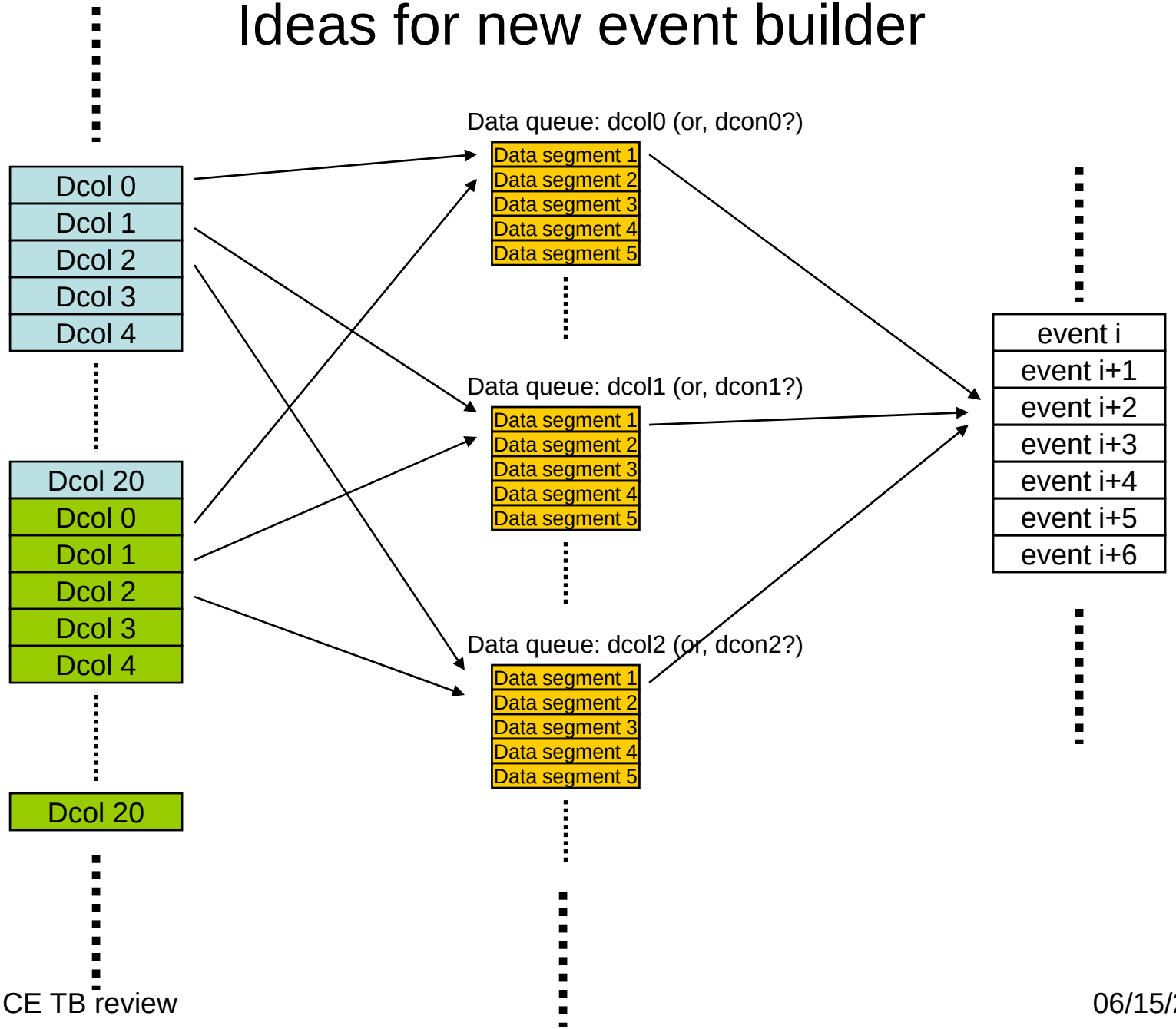
1. Time stamp
2. DCON trigger package
3. Neighboring packages
4. Known error features

# Projected 1m$^3$ data structure




Data segments from each dcol will be larger
May have a lot of triggers
Event boundary maybe violated very often
Data segment may end in the middle of an event
…

# Ideas for new event builder



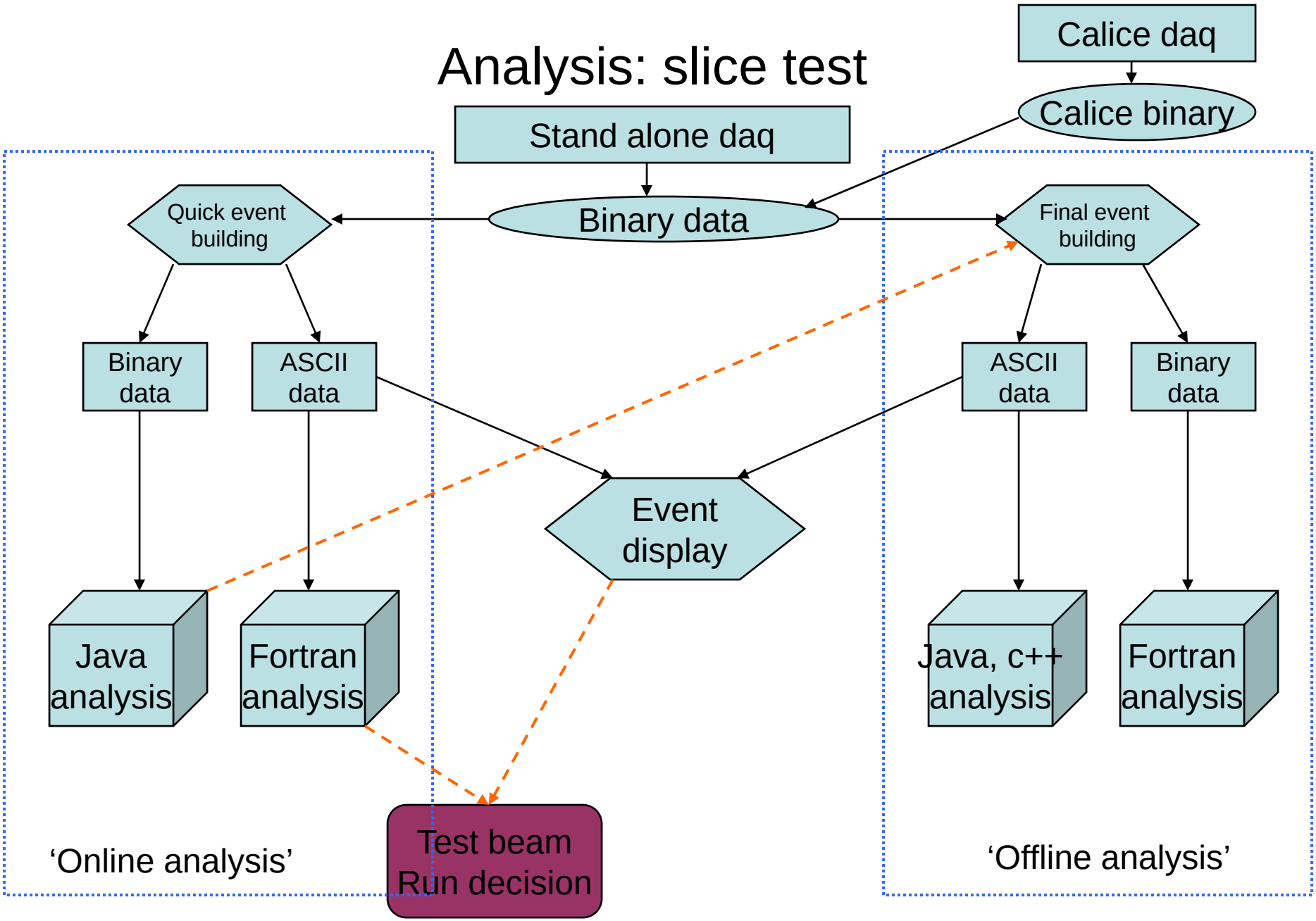
# 1m3 event builder: data error

- All current data error should be considered in the event building initially
  - Some could be removed later
- Need to be prepared for new error types
- Due to larger event size, error rate will be higher
  - Error rate: N(event with error)/N(all events)
- This is the major challenge for getting a reliable event builder
  - Progress will be made over time, as we gain experience with new readout

# CALICE lcio conversion

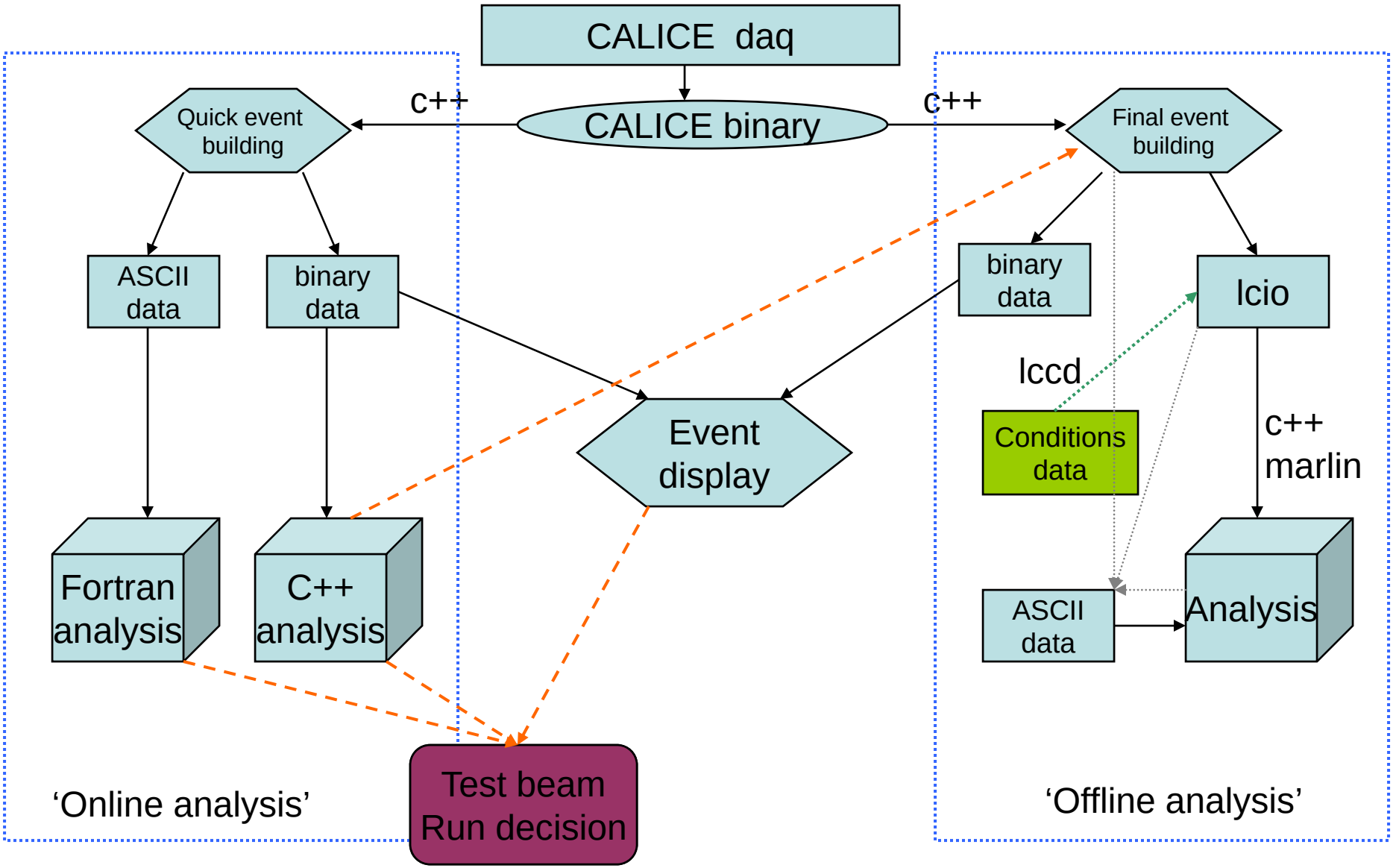
- We will provide lcio converter an event building algorithm – this is essentially our event builder
  - Input: CALICE daq binary (not our current converted binary) – this should be our goal anyway (for online analysis)
  - Output: event (format?)
  - Need to work with CALICE daq package: implies programming language to be c++ (currently java)

# Analysis: slice test




Conditions data was recorded by hand in log book

# Analysis: 1m$^3$



# Event display

- George Mavromanolakis developed the event display for slice test
  - Based on CALICE online display program
  - Read in ASCII data format
  - Geometry based on slice test setup
- Ben will extend the program for $1m^3$ tests
  - Input will be some kind of binary format, after event building
  - Geometry need to be extended to $1m^3$ setups

# Noise runs

- Regular noise runs (~5mins, self-trigger) generate a lot of data
  - Current rate: ~1-2MB/~2k chs/run
  - 1m3: 200-500MB/run, if RPC reaches slice test quality
- Cosmic ray calibration may come out of noise runs as well
  - Data size would be prohibitive, without online filtering
  - To first order, some kind of simple event building is needed within daq, in order to do the filtering
    - How robust does it need to be?
    - How fast does it have to be, in order to be put into daq?
  - Suppose we have a fast event building imbedded into daq
    - Can this be robust/fast enough to be the final event building?
    - Can we assure a data stream recovery after online building?

# $1m^3$ simulation

- Current simulation involves two detached steps
  - Geant4: c++, simulate test setup, write out position of energy depositions in ASCII format
  - RPCsim: Fortran, simulate RPC charge distribution, electronic threshold and generate hits, again, write out hit positions in ASCII format
  - Slice test simulation has been very successful
  - Currently studying $1m^3$ setup
- At some point, need to merge into a single package
  - Use Mokka?
  - Use lcio output?