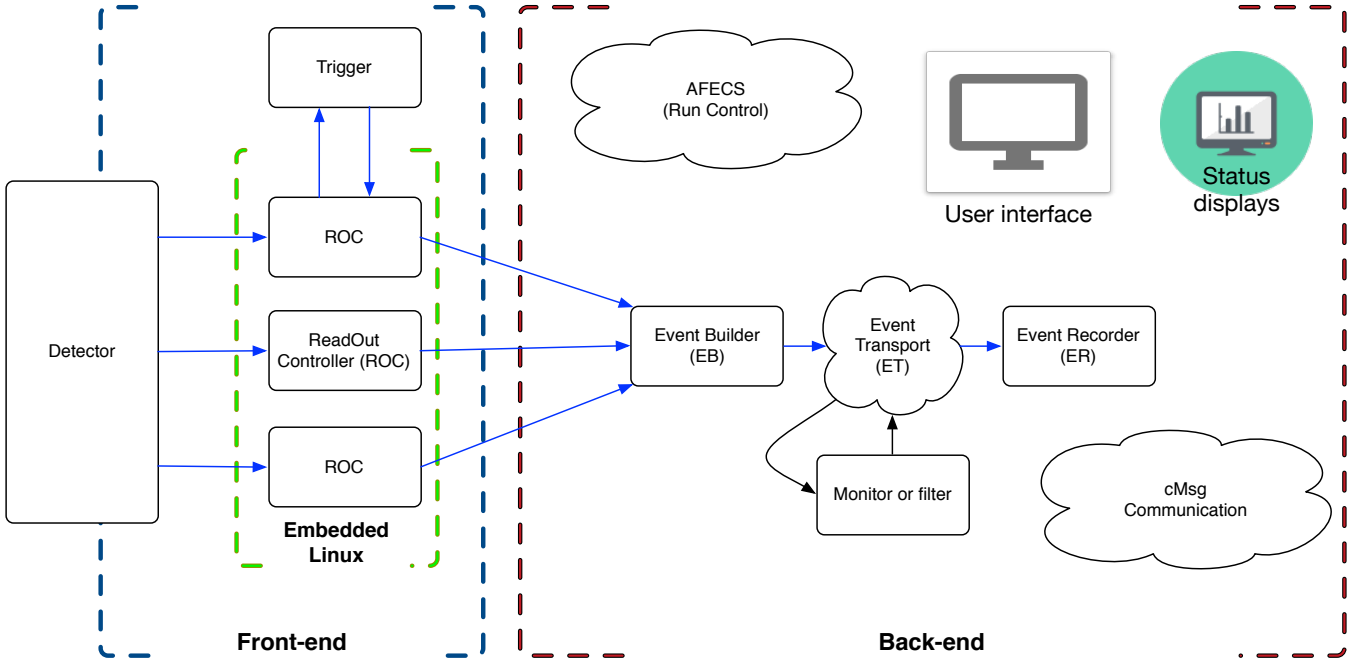
CODA Status
(not my title)

Graham Heyes

February 23rd 2018

CODA – as used by GlueX

- Front-end : Electronics, drivers, readout lists, ROC, Linux, Linux drivers.
- Back-end : Everything else
- Infrastructure : Third-party stuff, Linux, Drivers, network, RAID array(s).



Where are we?

- On Monday of this week we got halls B, D and DAQ group together for a chat.
  - Halls B and D have almost identical front-end.
    - Halls B and D see the same problems.
  - Hall B had a working DAQ at the end of 6 GeV running.
    - Current hall-B back-end is basically the same one used for 6 GeV with some modifications. They have no issues at the rates they have run at so far.
  - GlueX back-end has several as yet unresolved issues:
    - Stability, there are some strange issues that prevent runs from starting under some conditions.
    - Rate – as configured is fine for this run but not for Fall.
    - Lower priority bugs/implementation issues.

Front-end

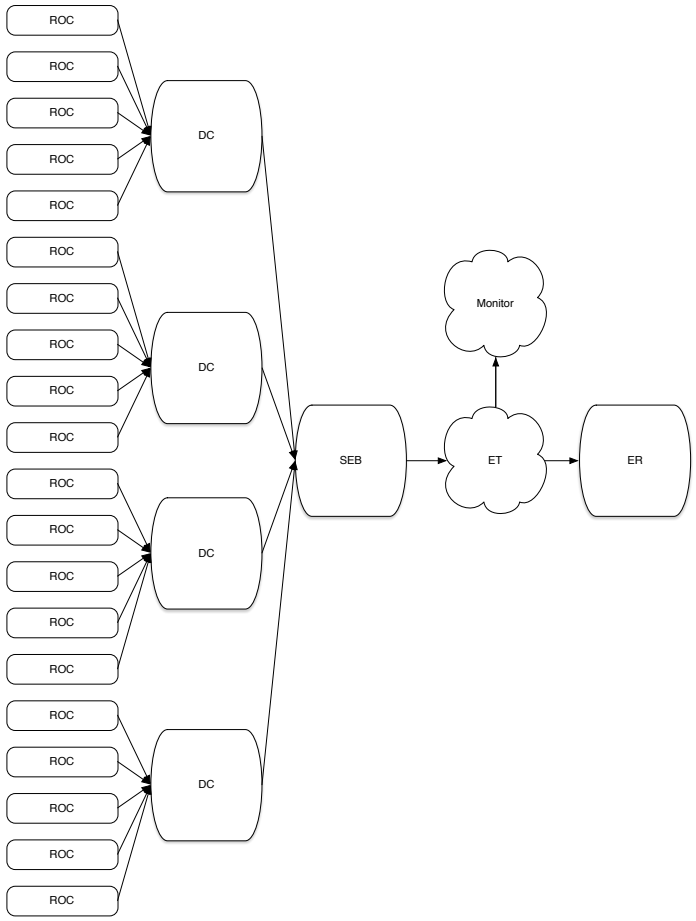
- Front-end issues account for lost beam time ~10% for each of halls B and D.
  - Some issues were a surprise in that, until now, it had not been made clear that they were a problem at the level that was discussed.
  - In particular there were phrases like “the well-known XYZ problem” where the problem was well known but only to those who knew it well, this is a communication issue.
  - Beam time this year is limited but intermittent and hard to diagnose issues take time to fix properly – some things not fixed.
- Path forward:
  - Both halls will provide a clear prioritized list.
  - Both halls + DAQ + Fast Electronics will work on communication!

Back-end : Run control

- Run Control is very stable and issues fall in three categories :
  - Failure to communicate with a ROC, EB etc is frequently perceived as a Run Control problem so RC is needlessly restarted!
  - Front-end issues generate errors that are MEANT to prevent a run from being started but this is perceived as a Run Control problem.
  - Starting a run can take a long time. This is not a problem with Run Control but with other parts of the DAQ system, in particular the distributed EB.

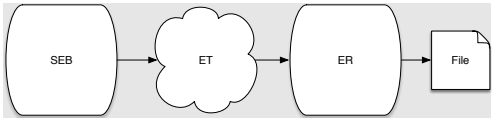
Back-end: Event Builder

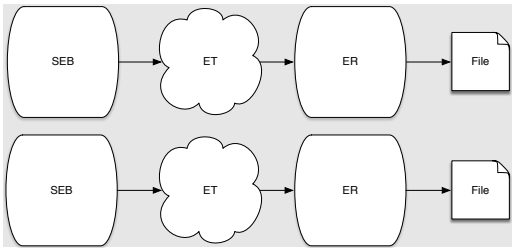
- Sometimes the initial DC to EB network connection takes a long time. It doesn't fail it's slow!
  - Simplify code as much as possible.
- DCs don't play well together on the same machine.
- ROCs per DC limited.
  - < 12 ROCs per DC runs fine.
  - >= 13 ROCs per DC is much slower.
  - We don't have enough real ROCs to test in the lab – need your system.
- Current production system has a single SEB and ER.
  - Rate limit is the ER writing a single file.

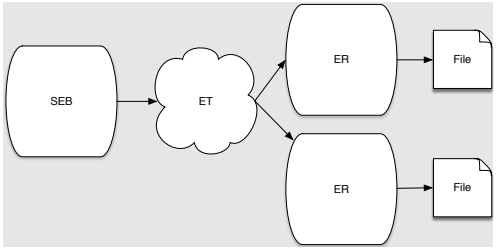


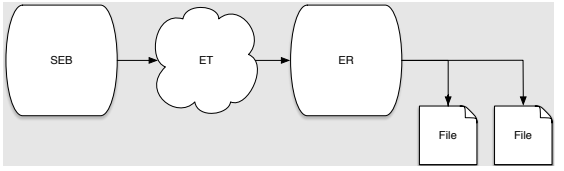
Back-end: Improve file writing

- One file limits us to 900 Mbyte/s
- Could use:
  - Two SEB, two ER and two files.
  - One SEB, two ER and two files.
  - One SEB, one ER and two files.
- Issue is that ER needs to see the END event from the SEB to close the files and end the run correctly.
  - Two ERs requires duplication of control events.
- Bottom solution is preferred for these rates but involves changes to ER and/or EVIO.
- We are working on this.
  - Need time with the GlueX system when ready!!









Back-end : cMsg

- We wrote cMsg because, at the time, no other freely available messaging system met the requirements.
  - Used in the background for CODA components to communicate with each other and with Run Control.
  - Was never intended to be a high data volume, high message rate general purpose messaging system.
  - Just fine for low rate small messages.
  - Users should use xMsg, used by CLARA, which is a layer on top of the open source zeroMQ package, instead of cMsg for high data volume applications.

What next? – not quite conclusion

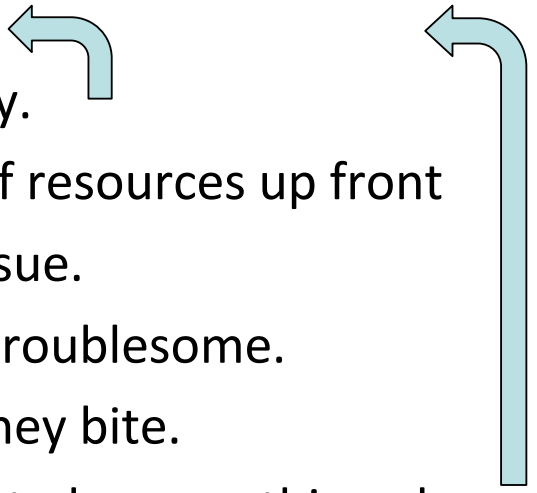
- GlueX and CLAS12 will provide prioritized lists of front-end issues.
  - DAQ and Fast Electronics will work aggressively on permanent solutions.
  - We (DAQ and FE) will likely require access to hardware with and without beam.
- Back-end issues are known but the causes of some of them are not well understood.
  - Need testing time or at least diagnostic feedback.
  - Rate issues do not appear to be show stoppers given the current system performance.

Mutterings – CLAS12

- All the woes of the world will be solved if GlueX uses the CLAS12 back-end…
  - The CLAS12 system is a “hacked/modified” version of CODA 2.5.
  - The EB is single threaded and single process, it has inherent bottlenecks and not much room for improvement.
  - The GlueX front-end data is not in a compatible format.
    - In particular the trigger data is not handled correctly.
  - The output of the CLAS12 EB is not in a format compatible with the GLUEX monitoring and offline.
- It is much more work to take the CLAS12 code and retro fit it than to fix the remaining CODA 3 issues OR to rewrite the CODA EB and ER using techniques that were in their infancy when they were designed.

Mutterings - Java

- We'd be better off without Java in the EB and ER…
  - Maybe true but not for the reasons that you would think.
  - The stability and performance of the Java code is no worse than C or C++ code that does exactly the same job.
  - The lessons learned are:
    - The Java VM is a black box - many moving parts outside our control.
    - Debugging can be a challenge.
    - Working in the OO paradigm can add complexity.
    - Because code runs in a VM the VM grabs a lot of resources up front
      - Multiple VMs on the same machine are an issue.
      - Linux's response to "badly behaved VMs" is troublesome.
    - Java is very forgiving, issues take while before they bite.
  - Many issues blamed on the use of Java turned out to be something else.



Back room developments

- Data compression
  - Working on an EVIO that compresses the data payload between uncompressed block headers.
    - 900 Mbyte/s disk limit relaxes.
  - Problem is that single thread compression limits to 300 Mbyte/s compression rate and multi thread version is complex.
  - Bought a commercial compression accelerator
    - Multi Gbyte/s rates but needs EVIO support.
- zeroMQ – open source library under xMsg used by CLARA would allow rapid development of a much simpler EB and ER with C/C++ as programming language – remove the VM from the data path.
  - What sort of priority should this be given?
  - **Get things running stably at Fall run rates first.**