Impressions of DAQ -1 Software

Murrough Landon – 8 December 1999

DAQ -1 Project

- (Pre)Prototype of the final ATLAS DAQ system
- Two major components:
- Dataflow: fast data transport (specific solution to a specific problem)
- Back end: slow support software (general solution to general problem)
- Basic requirements: distributed, partitionable, configurable by databases

Summary so far

- It exists and was tested on a small system in June/July
- Dataflow software not currently suitable for use in a smaller scale environment such as ROD crates
- Back end software components are now distributed on CDROM

Impressions of Back end software

- Its a prototype: rough edges, evolving, incomplete
- It consists of many components, some quite complex
- On the whole it is well documented
- We can use it

Back End Components

Overview

- Core components include: configuration databases, error message reporting, interprocess communication, process managements, run control.
- Other components include: integrated GUI, test and diagnostics managers, online bookkeeper, etc
- All the core components have been evaluated and I have tried running their test DAQ system via the integrated GUI.
- Support from the DAQ group by email has been helpful (and necessary).

Back End Details (1)

Configuration Databases

- Detailed object database schema
- Low level database editors (cumbersome to use)
- Extensive documentation
- Learning curve for OO database ideas....
- Data access library (exists) and “partition editor” (coming) are layered on top of low level software and know about the structure of the database.
- These will need extending for subdetector specific classes

Message Reporting Service (MRS)

- Very easy to use (iostream like API)
- Good documentation
- Doesnt require use of the database

Information Service (IS)

- Easy to use (could be better)
- Good documentation (lacking good intro)
- Doesnt require use of the database
- Could provide “poor mans” persistent database

Back End Details (2)

Process Manager (PMG)

- Start/stop/control distributed processes
- Documentation somewhat out of date
- Doesnt require use of the database
- But does seem to need whole DAQ infrastructure

Run Control (RC)

- "Concurrent Hierarchical State Machine" (CHSM)
- Very good documentation: extensive description, tutorials, reference, examples
- Fully controlled by the database
- Requires the whole DAQ infrastructure

Integrated GUI (IGUI)

- Newish component implemented in Java
- Combines various expert GUIs with "shift" overview of the DAQ
- Needs to be customisable by subdetectors (eg calibration runs, subdetector specific displays etc) $\rightarrow$ java beans?

Dataflow Software

Overview

- Present system requires several LynxOS processors, each emulating a component of the final Readout Crate: ROBs, trigger module, event builder interface, crate controller (LDAQ), etc
- We require a system running in a single processor
- They have started to collapse their system into a single processor but more work is required
- Also the design requirements in the ROD crate may be different

Timescales

- Until April 2000, all their effort is concentrated on the L2/DAQ technical proposal - ie agreeing what a readout crate will look like
- At least three months work (probably needing help from subdetectors) is anticipated to produce a dataflow solution suitable for ROD crates
- Some subdetectors want a DAQ -1 style test beam DAQ for summer test beams...but this may be optimistic

Summary

- We can use the back end software now...
- ...though it is overkill for reading a few events from a single ROD (starting a run requires at least ten processes)
- In the short/medium term we will need our own dataflow type software
- Some work has started on providing components for this (ROOT) and on an overall framework
- Later we would have to migrate this to the final ROD crate DAQ
- Alternative scenario is to continue with our “old” DAQ software and incorporate DAQ -1 components slowly...
- In either case, considerable effort will be required for designing and implementing our configuration database and associated software
- Also for prototype calibration and monitoring procedures...
- We must aim to design our software to be as independent of its evolving environment as possible