Chlebana
Nov 24 1999

DAQ Status

Run Control

Front End Readout

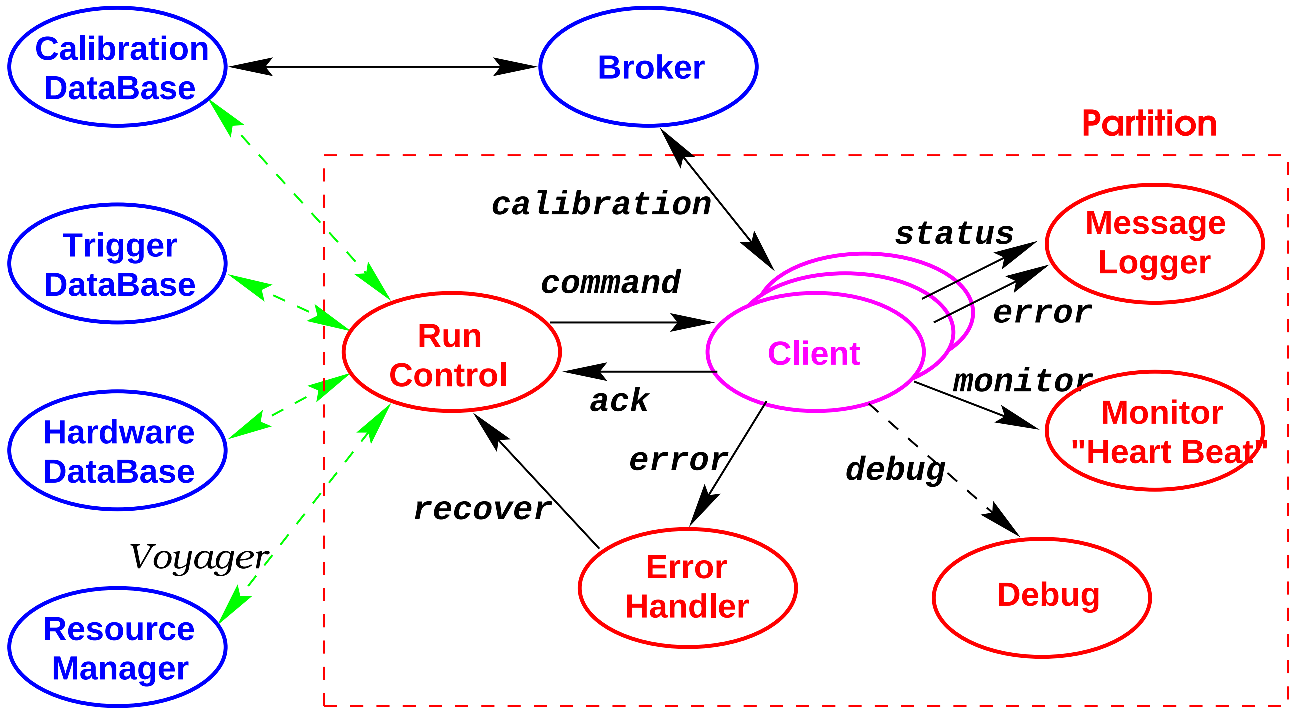
DAQ Monitoring

Data Monitoring

Trigger

Run Control

We have had a series of meetings to specify the functional requirements for Run Control and have specified the communication framework.



Commercial product SmartSockets used for communication.

On top of that we added a layer (DAQMSG) that makes it easy to pack messages in the required format and to synchronize message formats between the server (Java based) and client (C or Java).

We are also using the Merlin facility for messages that need to be logged.

Most of this communication network is specified and we have working examples of the different parts.

We've met an internal milestone of readout three components which represent the major parts of the detector.

ADMEM	(Calorimeter)
TDC	(COT and Muon)
SRC	(Silicon)
SMQIE	(Shower Max)

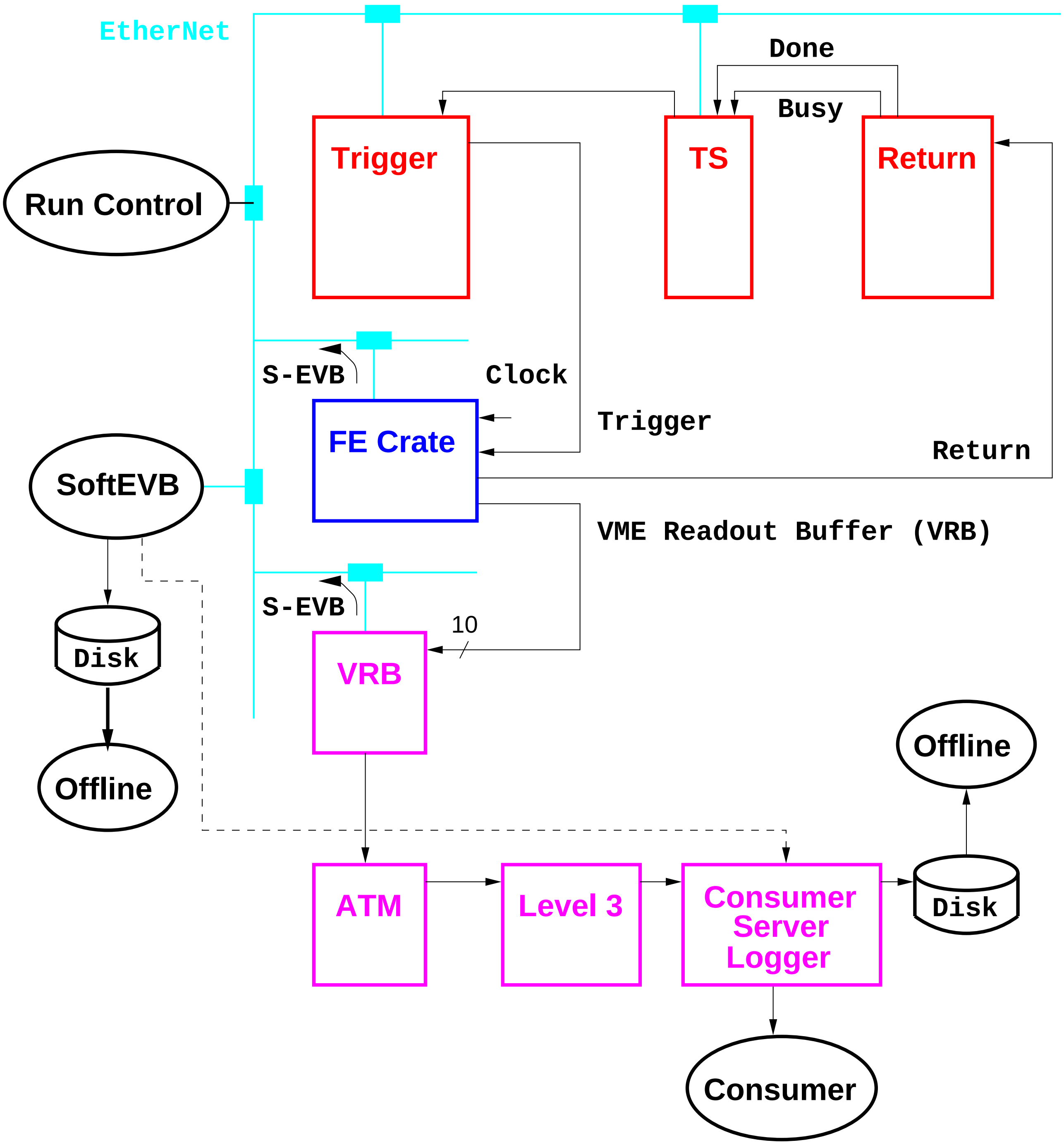
Separately we've readout trigger cards. Readout and formatting of the trigger card data is similar for this class of cards allowing us to use the same code...

Run Control was used to start the run:

- Send Transition commands to the clients
- Send Configuration information to client
- Wait for acknowledgments

Trigger signals were initiated by the Trigger Supervisor and distributed to the Front End crates.

Tested software to transport data through the entire chain.



The "Three Component" test represent a major test of the software and hardware.

We ran for about 12 hours at a trigger rate of 33 Hz, 1.4 million triggers were taken.

Triggers generated by the Trigger Supervisor
$\rightarrow$ Calorimeter trigger will soon be available.

Trigger signals fanned out through the trigger cross point to three components.

Silicon Readout Control distributes trigger to the FIB crates.

Return signals sent back to TS through the return cross point (used to hold off triggers until all components are done).

Data was sent up to the VME Readout Buffers.

Some formatting done in the front end crate.

Readout the VRB by the Event Builder processors.

Transports data through ATM switch.

Data fragments were assembled into the final event format.

Sent events to the Level 3 processors.

Level 3 filter was run to select events based on calorimeter energy.

Events were passed to the Consumer Server/Logger which stored them on disk and sent events to a consumer process.

Were able to display histograms as data was collected.

This full chain test is *very* close to the real system...

For this test we used configuration files to setup the clients.
→ Eventually information will be extracted from databases.

→ System experts need to specify the configuration message.

At the start of a run we will access the resource manager to select a partition (one of eight trigger supervisors) and book clients. → tested in a standalone version and is being integrated into run control.

We have constructed the configuration message from the database but at this point it is still more flexible to use configuration files.

When starting the run we still need several experts to start up the different parts.

→ Communication with the Event Builder will go through a proxy which exists. Had a first meeting to discuss communication between Run Control and the proxy. Expect to work on this next week.

→ For now the Consumer Server/Logger is always running. We plan to be able to start and stop the CS/L from Run Control. Code exists, just have to agree on some details of the message format.

Don't see any technical barriers to adding additional components.

In Addition...

We routinely run with multiple partitions.

Several Trigger Supervisors sending trigger signals to different clients.

We ran one partition with the hardware event builder and a second partition sending data to the software event builder.

Have also run using three Trigger Supervisors (all that was available at that time)

Have used the calibration broker to download Silicon calibration data.

Software Event Builder

We can bypass the real event builder and send data from the front end crates to a software event builder.

We can also send formatted data from the software event builder to the consumer server logger.

Can be used as a DAQ without the hardware event builder.

We are also able to send data from the software event builder to the Consumer Server/Logger.

The software event builder makes use of the offline “Banks” package.

As more components are read out we need to add support code to the Banks package.

→ Need to identify a person to work on updating and maintaining the software event builder.

→ Need to have someone to update and maintain the “Banks” package.

Monitor

We can split monitoring into two parts

DAQ monitoring

Make sure that the data flows through the system and identify why a run stopped.

$\rightarrow$ Experts will specify what information is useful to monitor and provide the actual code.

We want to have a high level panel that displays the status of a sub system and by clicking on a button for that subsystem you will call up more detailed windows.

We have a working example that monitors the status of the front end processors $\rightarrow$ subsystem monitoring should follow this example

Data monitoring

Consumers will monitor data quality, trigger etc...

Users can write consumer modules to access data as it is being collected. This framework is well developed.

Have run consumers during data taking to plot histograms of pedestals.

$\rightarrow$ work is needed to specify what quantities should be monitored (reference histograms etc...).

Calibration

Run Control framework used to book a partition select clients. Used to initiate the calibration.

State manager is flexible and we can easily specify a new state diagram for particular types of operations (data taking, calibration, etc...).

Standalone code can be developed without the run control framework.

Can then call calibration routine from run control allow us to calibrate multiple crates at the same time and send up calibration data to be stored in a database.

Several types of calibration are being developed in this way, QIE, COT, Source...

Calorimeter calibration data has been sent to a software event builder then sent to a calibration consumer which stores the results in a database.

$\rightarrow$ Starting to exercise full chain.

$\rightarrow$ Take regular calibrations.

$\rightarrow$ Test code to examine contents of the database.

$\rightarrow$ Requires some work on the software event builder...

Trigger

Starting to integrate the Calorimeter Trigger.

Energy sums used to create a Level 1 trigger.

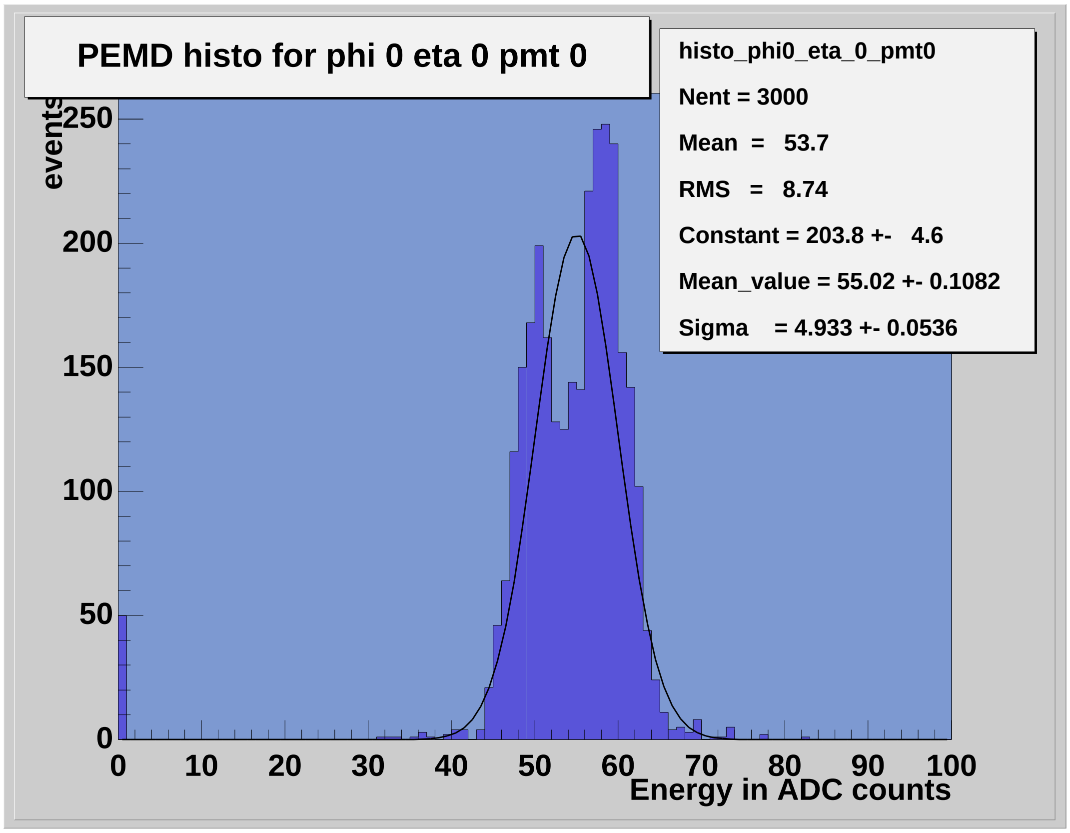
Trigger crates are currently powered off in order to pull cables.

Very close to having a real Cosmic Trigger.

Will resume integration when power can be turned back on.

In the mean time can work on configuring the trigger cards and readout of the trigger information.

Histogram from a Consumer process.



Double peak due to cuts at Level 3.

Entries in bin 0 is from an uninstrumented channel.

Short Term Goals

Need to work on being able to reliably start a run from one place.

- Access the resource manager to select a partition
- Fill information in database
- Construct configuration message from database access
- Run Control communication with the Event Builder
- Run Control communication the Consumer Server/Logger

Get the Cosmic Trigger working

Configuration messages need to be defined for additional clients, trigger cards...

Readout and formatting trigger banks.

Additional support for calibration runs. → update software event builder

For the mock data challenge the Level 3 will read MC data from disk files. Will not use the full chain...