artdaq: DAQ Software Development Made Simple

John Freeman

CHEP 2016

10 October 2016

The “*art*” in *artdaq*

- “*art*” is an application developed in Fermilab’s Scientific Computing Division which performs event-based processing for an experiment’s offline analysis
- This processing is done using pluggable modules; modules can perform event filtering, analysis, reconstruction and output
- A standard set of modules is available + experiments can write their own
- The choice of modules is referred to as an *art* “workflow”, and is communicated to art via a FHiCL (*.fcl) document (Fermilab Hierachical Command Language – think JSON, or XML).
- An example of running art at the command line:

```
art –s ExperimentsInputFile.root –c ExperimentSpecificWorkflow.fcl
```

For more, see art.fnal.gov

Motivation

- *art*'s features would be very useful for online running as well – events being produced in real time:
  - Filtering can reduce the data initially stored
  - Analysis allows for online monitoring
  - Modules could be written in common for offline and online
  - A DAQ could take advantage of existing modules
- This is where *artdaq* comes in!

artdaq Is

- A set of processes, which provide "hooks" for experiments to embed code (primarily *art* modules + communication with upstream hardware)
- Additionally, infrastructure for
  - State-machine DAQ transitions ("start", "stop", etc.)
  - Transport + assembly of data fragments
  - DAQ metrics reporting (event rate, etc.)
- FHiCL-configurable, like *art* – very flexible
- Supported for most major Linux variants
- A simple "toy" *artdaq*-based DAQ system will be described on the next few slides
  - Keep in mind when the system is described that a real-life system will have more of each type of process
  - Also keep in mind that the processes can (and probably will) run on different hosts

BoardReaders: Interface to the Hardware

TPC DATA

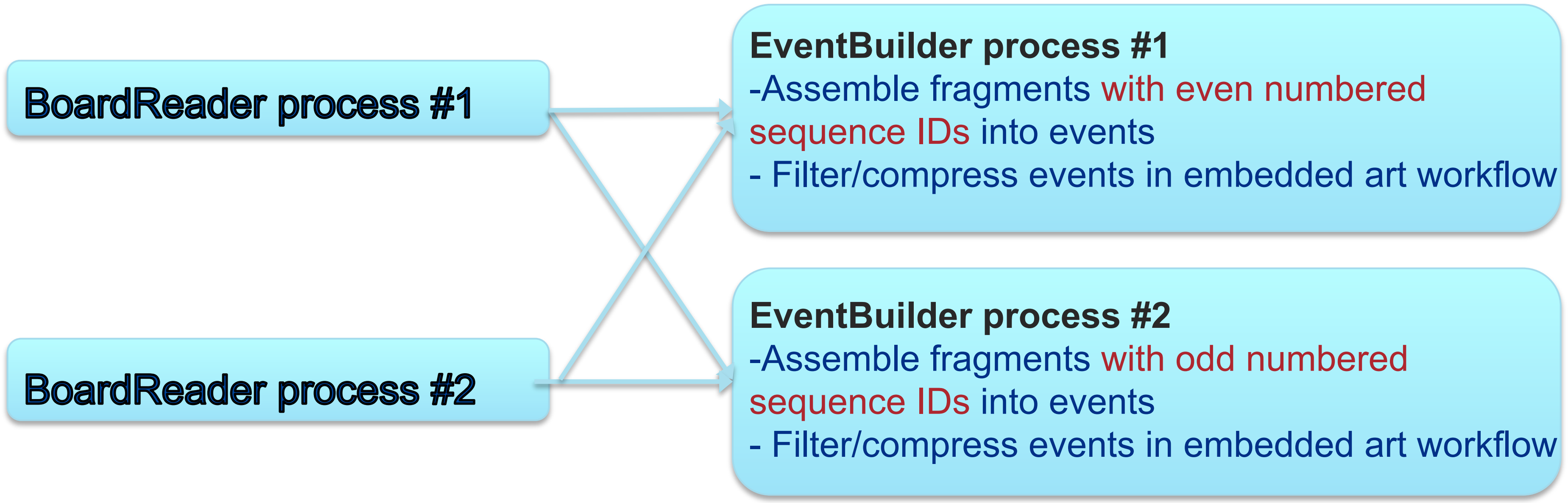
BoardReader process #1
Continuously call *ExperimentSpecificClass1::getNext* while running

PMT DATA

BoardReader process #2
Continuously call *ExperimentSpecificClass2::getNext* while running

- BoardReaders call objects (here, ExperimentSpecificClass1 and 2) which implement the ***artdaq::CommandableFragmentGenerator*** base class's functions – ***start, getNext, stop***
- ***getNext*** reads data in according to the experiment's protocol and returns it wrapped in artdaq::Fragment objects (data stamped with a fragment ID and sequence ID)

EventBuilders: Assembly and Filtering/Compression


BoardReader process #1

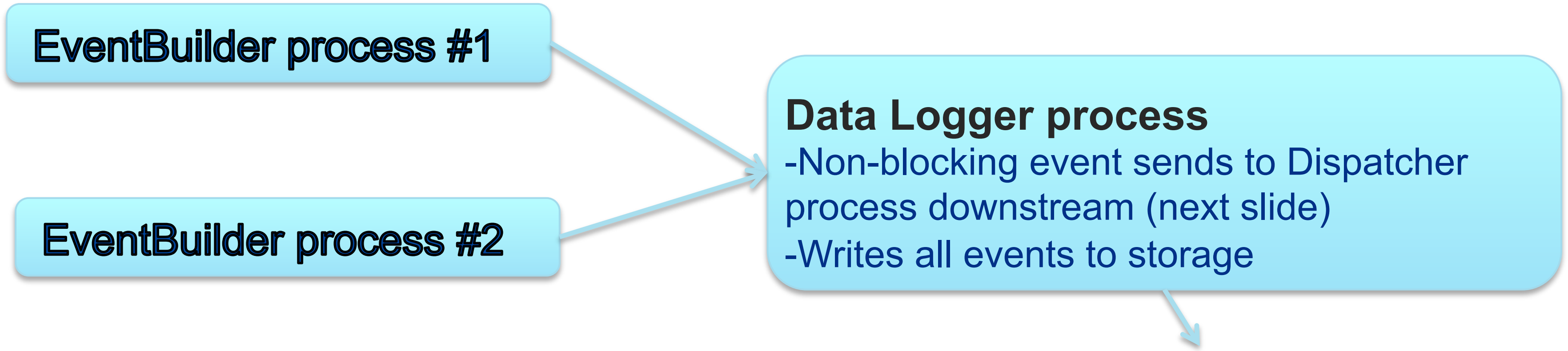
BoardReader process #2

EventBuilder process #1
-Assemble fragments with even numbered sequence IDs into events
- Filter/compress events in embedded art workflow

EventBuilder process #2
-Assemble fragments with odd numbered sequence IDs into events
- Filter/compress events in embedded art workflow

- "Round Robin" fragment sending:
  - Each BoardReader sends fragments with a fixed fragment ID, all sequence IDs
  - Each EventBuilder is in charge of assembling all fragment IDs for 1/N sequence IDs

Diskwriting



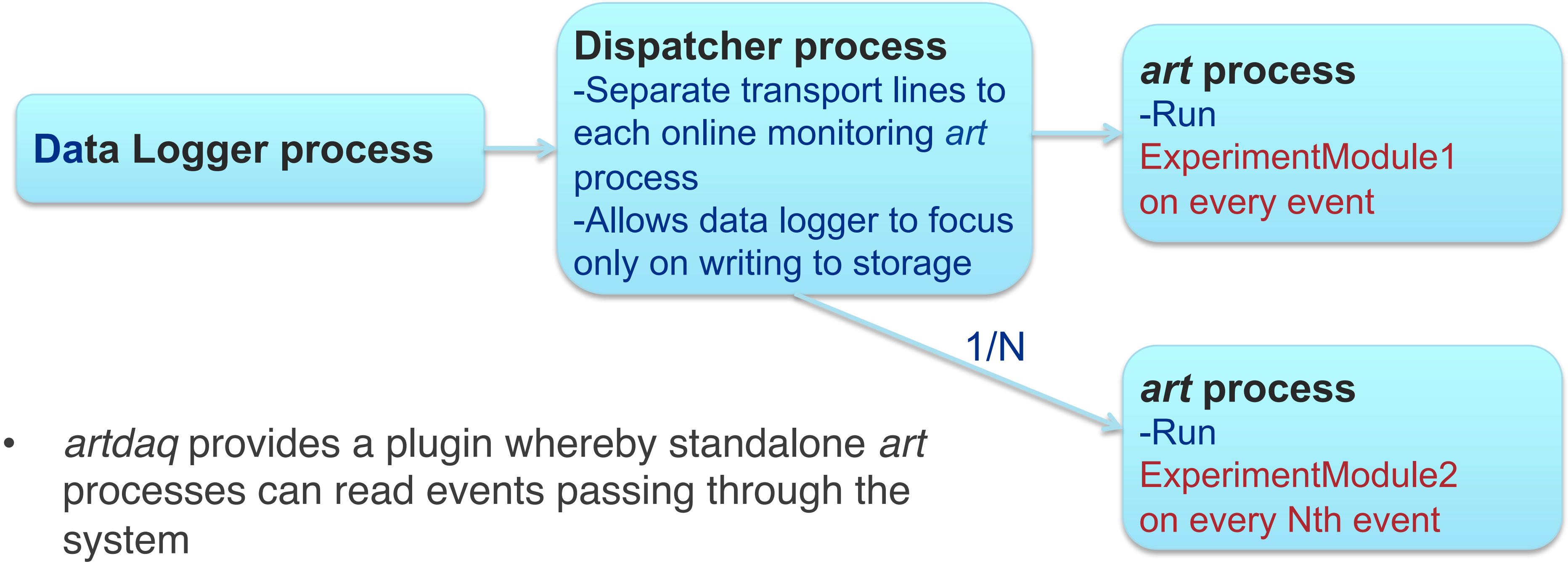
EventBuilder process #1

EventBuilder process #2

Data Logger process
-Non-blocking event sends to Dispatcher process downstream (next slide)
-Writes all events to storage

- Events are saved in *art*-readable *.root files
- The FHiCL documents used to configure the *artdaq* processes (and hence the DAQ) can also be saved in the *.root files

Online Physics Monitoring



Data Logger process
Dispatcher process
-Separate transport lines to each online monitoring *art* process
-Allows data logger to focus only on writing to storage
***art* process**
-Run ExperimentModule1 on every event
1/N
***art* process**
-Run ExperimentModule2 on every Nth event

- *artdaq* provides a plugin whereby standalone *art* processes can read events passing through the system
- Can configure fraction of events sent to a process, or even apply experiment-specific cuts!

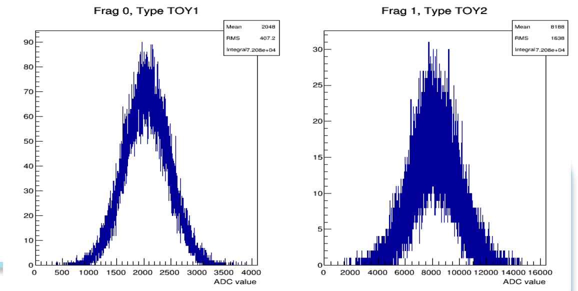
Online Physics Monitoring

Data Logger process

Dispatcher process

-Separate transport lines to each online monitoring *art* process

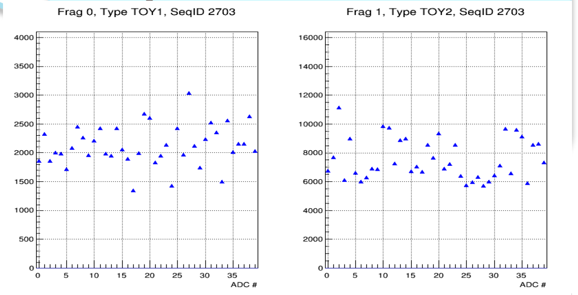
-Allows data logger to focus only on writing to storage



on every event

1/N

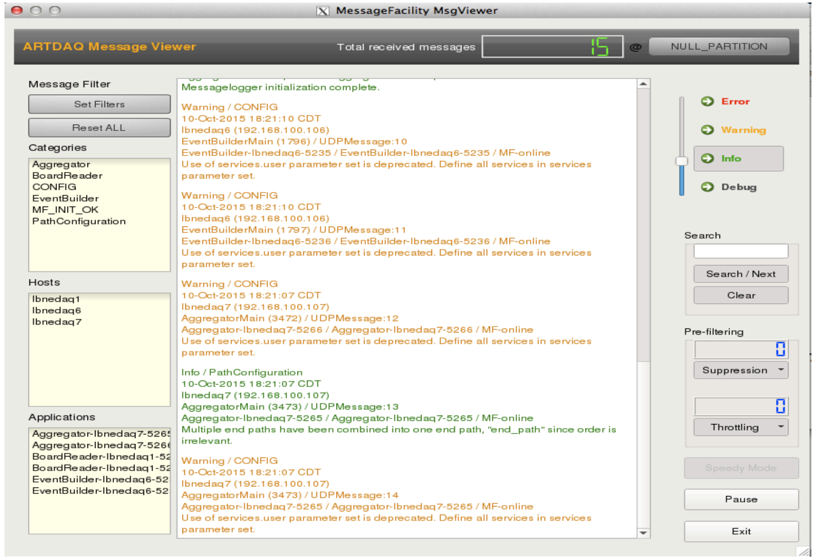
***art* process**



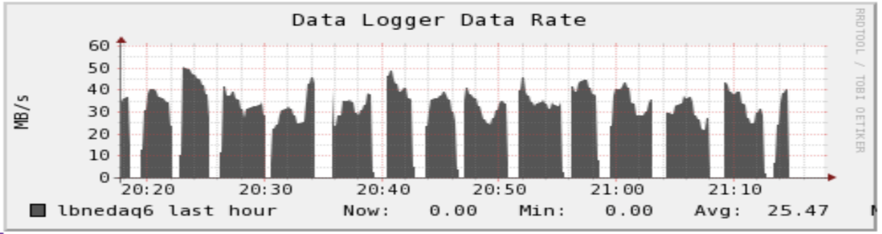
- *artdaq* provides a plugin whereby standalone *art* processes can read events passing through the system
- Can configure fraction of events sent to a process, or even apply experiment-specific cuts!

DAQ Monitoring and More

- *artdaq* provides the MessageViewer app, which prints messages from both *artdaq* and experiment-specific code with severity level indicated by color
- Plugins are provided so that the metrics reported by *artdaq* processes can be displayed in different formats (Ganglia, Graphite, etc. – FHiCL configurable)
- TRACE debugging



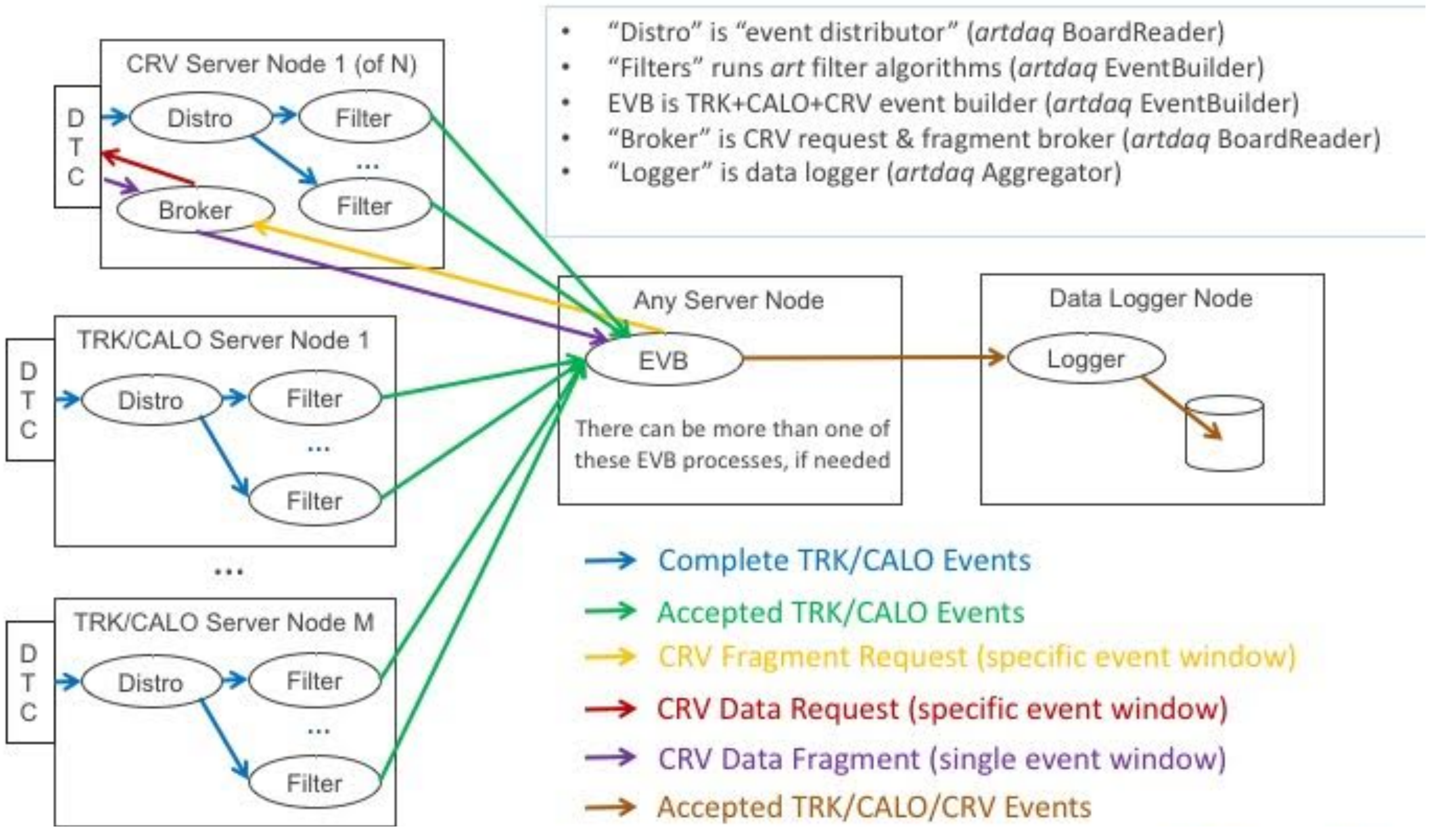
```
Messagelogger initialization complete.

Warning / CONFIG
10-Oct-2015 18:21:10 CDT
lbnedaq6 (192.168.100.106)
EventBuilderMain (1796) / UDPMessage:10
EventBuilder-lbnedaq6-5235 / EventBuilder-lbnedaq6-5235 / MF-online
Use of services.user parameter set is deprecated. Define all services in services
parameter set.

Warning / CONFIG
10-Oct-2015 18:21:10 CDT
lbnedaq6 (192.168.100.106)
EventBuilderMain (1797) / UDPMessage:11
EventBuilder-lbnedaq6-5236 / EventBuilder-lbnedaq6-5236 / MF-online
Use of services.user parameter set is deprecated. Define all services in services
parameter set.

Warning / CONFIG
10-Oct-2015 18:21:07 CDT
lbnedaq7 (192.168.100.107)
AggregatorMain (3472) / UDPMessage:12
Aggregator-lbnedaq7-5266 / Aggregator-lbnedaq7-5266 / MF-online
Use of services.user parameter set is deprecated. Define all services in services
parameter set.

Info / PathConfiguration
10-Oct-2015 18:21:07 CDT
lbnedaq7 (192.168.100.107)
AggregatorMain (3473) / UDPMessage:13
Aggregator-lbnedaq7-5265 / Aggregator-lbnedaq7-5265 / MF-online
Multiple end paths have been combined into one end path, "end_path" since order is
irrelevant.

Warning / CONFIG
10-Oct-2015 18:21:07 CDT
lbnedaq7 (192.168.100.107)
AggregatorMain (3473) / UDPMessage:14
Aggregator-lbnedaq7-5265 / Aggregator-lbnedaq7-5265 / MF-online
Use of services.user parameter set is deprecated. Define all services in services
parameter set.
```



Experiments Which Use *artdaq*

Experiment	Peak Incoming Data Rate (GB/s)	# BoardReaders	# EventBuilders	EventBuilder data reduction factor
DUNE 35ton	0.1	24	16	1
Darkside-50	0.5	12	16	~5
LArIAT	0.3	1	1	1
Mu2e	33	36	~500	~100
protoDUNE-SP	3	~80	10-20	1
SBND	0.4	~20	10-20	1
ICARUS	0.4	~20	10-20	1

Mu2e Planned Layout



- "Distro" is "event distributor" (*artdaq* BoardReader)
- "Filters" runs *art* filter algorithms (*artdaq* EventBuilder)
- EVB is TRK+CALO+CRV event builder (*artdaq* EventBuilder)
- "Broker" is CRV request & fragment broker (*artdaq* BoardReader)
- "Logger" is data logger (*artdaq* Aggregator)

→ Complete TRK/CALO Events

→ Accepted TRK/CALO Events

→ CRV Fragment Request (specific event window)

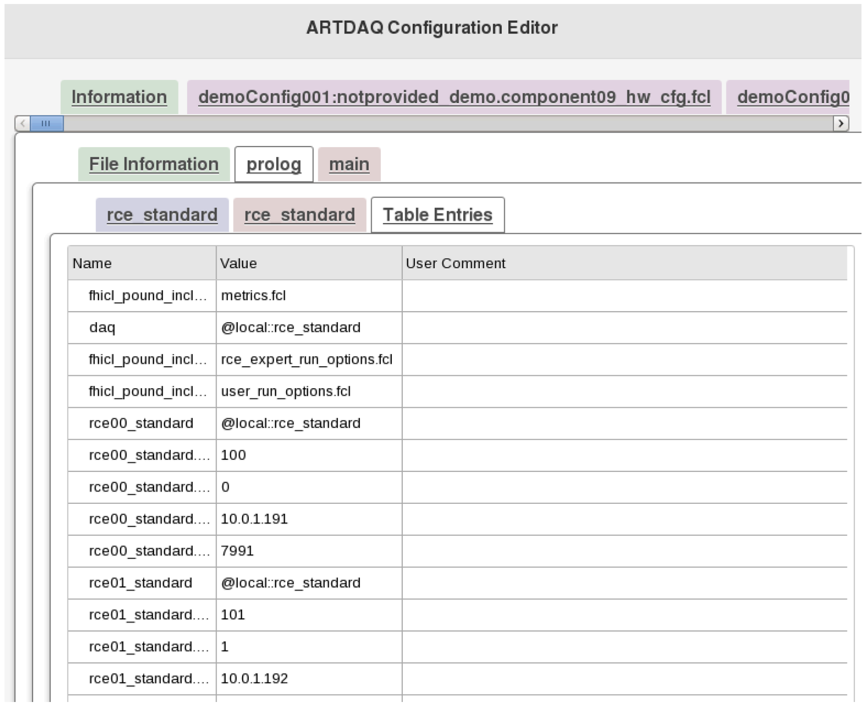
→ CRV Data Request (specific event window)

→ CRV Data Fragment (single event window)

→ Accepted TRK/CALO/CRV Events

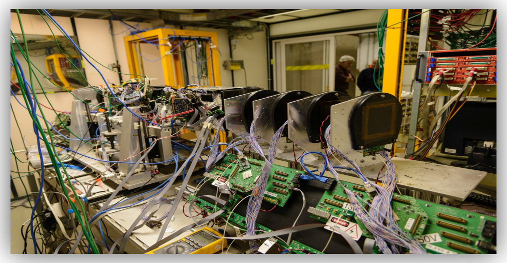
Upcoming Developments

- Convenience and choice
- Ability to configure FHiCL parameters via a GUI rather than through editing ASCII files
  - Can save/retrieve parameters in DB
- Run control / process management
  - Experiments won't need to develop software to control when *artdaq* processes are created, destroyed, and sent state transitions
- Data transport flexibility via plugins
  - Current data transport done via MPI
  - We'd like the transport layer to be something you could choose

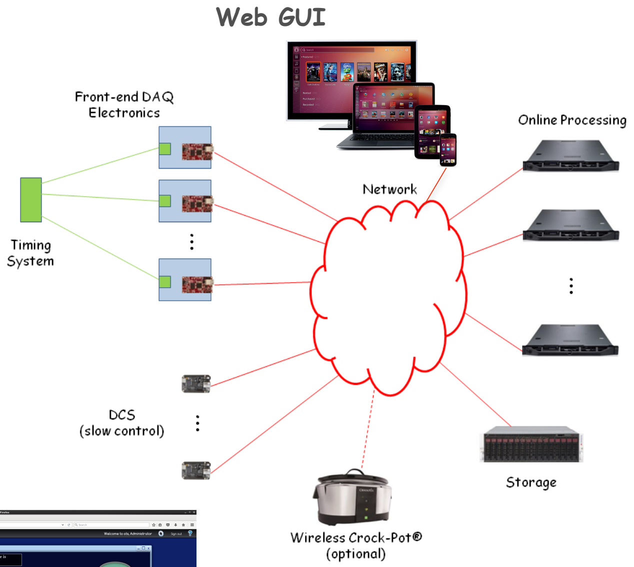


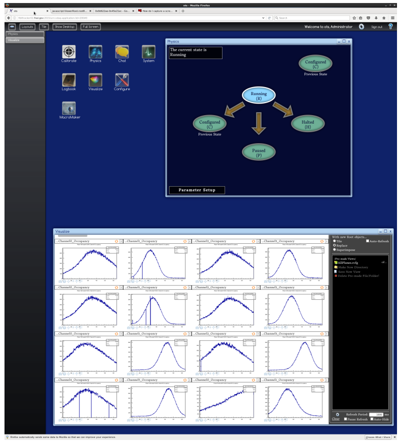
Name	Value	User Comment
fhicl_pound_incl…	metrics.fcl	
daq	@local::rce_standard	
fhicl_pound_incl…	rce_expert_run_options.fcl	
fhicl_pound_incl…	user_run_options.fcl	
rce00_standard	@local::rce_standard	
rce00_standard.…	100	
rce00_standard.…	0	
rce00_standard.…	10.0.1.191	
rce00_standard.…	7991	
rce01_standard	@local::rce_standard	
rce01_standard.…	101	
rce01_standard.…	1	
rce01_standard.…	10.0.1.192	

otsdaq



- *artdaq*-based DAQ toolkit
- Goal is to provide "off-the-shelf" DAQ components
- Designed for small lead-time experiments – get a DAQ up and running in a matter of hours
- Provides Run Control GUI, firmware for supported boards and configuration management system





http://otsdaq.fnal.gov/beta

Conclusions

- Developed by Fermilab's RSI (Real-Time Software Infrastructure) group, *artdaq* is used by many experiments
- Designed to provide online users the benefits of the *art* package, it also provides numerous useful features which experimenters won't need to build from the ground up
- *artdaq* was created to make experimenter's lives easier, and is constantly being improved with that goal in mind- *reusability and flexibility*
- To learn how to begin running a simple *artdaq*-based system within minutes, go to *https://cdcvs.fnal.gov/redmine/projects/artdaq-demo/wiki*
  - Works on most major Linux distributions (Scientific Linux, Ubuntu 14, …)
  - Can also run it out of VirtualBox, using this file: https://goo.gl/OoU6vJ