ProtoDUNE SP DAQ Dataflow Software

Edited by K. Biery (FNAL SCD/RSE)

1 Introduction

The dataflow software for the protoDUNE SP DAQ is primarily responsible for acquiring the data from the readout electronics, packaging it appropriately, and storing it in files that are local to the DAQ cluster. It has responsibility for other functions that include the delivery of configuration parameters to the front-end electronics and real-time monitoring of the quality of the data and the performance of the DAQ system. It is not responsible for the transfer of the raw data files to permanent storage.

2 Interfaces

One of the most important interfaces for the dataflow software is the computer, network, and storage hardware on which it runs. For the computers, this corresponds to the number of servers, the number of cores per server, and the amount of memory on each server. Insufficient processing power or memory will adversely affect the performance of the DAQ software. For the network, the important parameters include the speed of the networking equipment and the amount of buffering that it provides. For the storage systems, the important parameter is the rate at which data can be both written and read from the media. The read/write nature of this requirement is based on the need to support transfers of data files from the DAQ cluster to permanent storage while continuing to write new files.

Another important interface is to the Linux operation system that is running on the DAQ computers. One example of this is the appropriate tuning of the Linux disk cache so that disk-writing performance is not adversely affected.

The interface to the detector electronics includes the need to configure the electronics and read out data. There are interfaces to the run control and configuration management systems for the receiving and processing of transition requests and configuration data.

There is the extra interface to the Trigger System that allows the dataflow software to feed back information that can be used to reduce the rate of triggers, if needed.

At the back end of the system, there is the interface between the infrastructure that runs the real-time data quality monitoring algorithms and the algorithms themselves.

3 Requirements

The requirements of the DAQ that are related to the dataflow software include:

- Supporting data throughput and logging of 3 GB/s.
- Supporting the configuration and readout of 50 RCE boards, 24 SSPs, 2-3 FELIX cards, and the trigger and timing subsystems.
- Internal communication to support trigger throttling.

Currently, there is no requirement to incorporate beam instrumentation data in the TPC/PDS data path.

4 Design

The design of the protoDUNE DAQ dataflow software is based on *artdaq*, which a data acquisition toolkit developed at Fermilab. There are many introductory presentations on

artdaq available on the web (please see a list of talks and posters, the Real-Time 2012 conference paper, and the recent CHEP presentation), and the introduction to the artdaq-demo is also a useful introduction to *artdaq*.

Figure 1 shows some of the components and functions that are part of *artdaq*. The shapes that have a green background are provided in the core functionality, and the ones with an orange background are developed by the experiment. The primary data flow is shown horizontally in the middle of the diagram. BoardReader processes are responsible for communicating with the detector electronics, EventBuilder processes are responsible for assembling complete events and optionally processing them in art, and Aggregator processes are typically responsible for logging the data and serving events to real-time DQM processes.

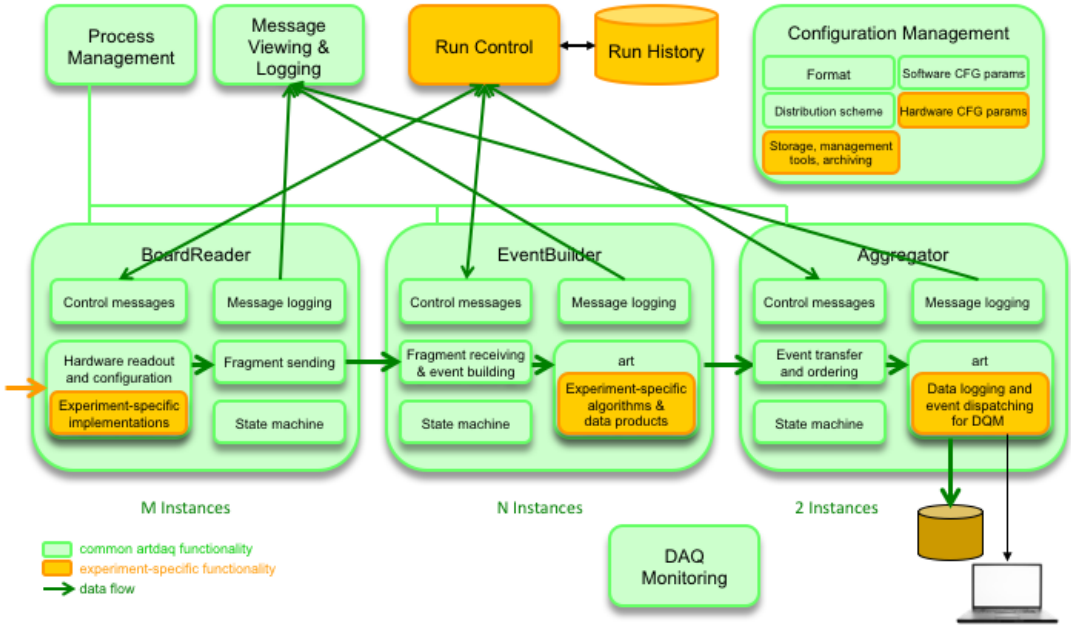

Figure 1 Components and functions that are part of *artdaq*.

Features of the protoDUNE DAQ system

Figure 2 shows the *artdaq* system that was used for 35-ton data taking, and Figure 3 through Figure 6 illustrate the changes that are being made for the protoDUNE DAQ.

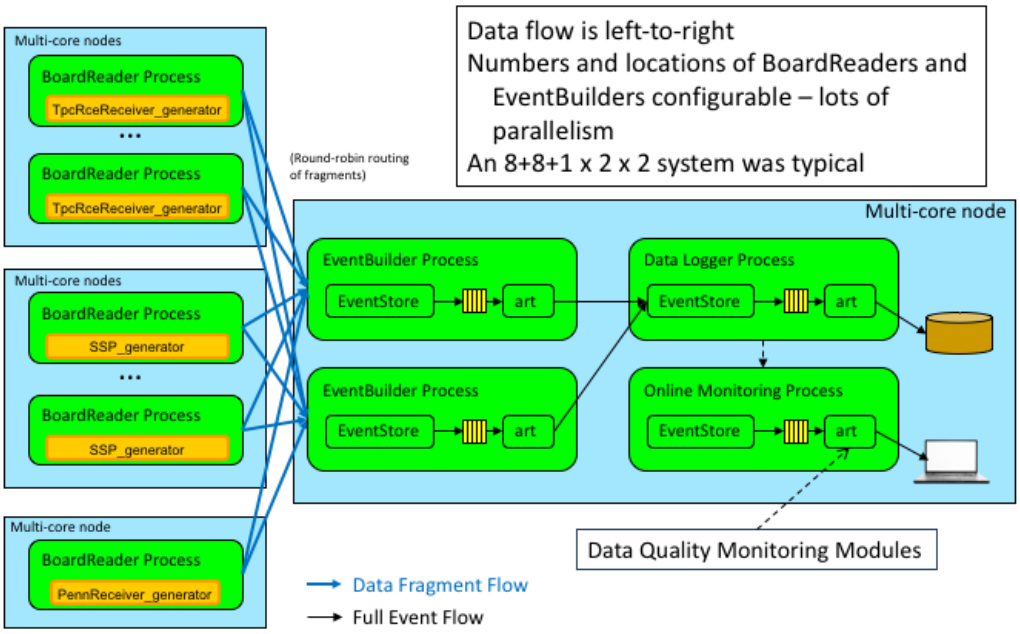

Figure 2 Diagram of the artdaq processes used in the 35-ton DAQ system.

Figure 3 illustrates the separation of the DQM processes from the main dataflow applications. Previously, the execution of the monitoring algorithms was rather tightly tied to the main applications. In the current version of artdaq, the DQM algorithms are run in stand-alone art processes that receive events via either shared memory or UDP multi-casts. In all cases, the data flow to the Dispatcher process and the DQM processes is non-blocking. That is, it can not affect the performance of the primary data flow.

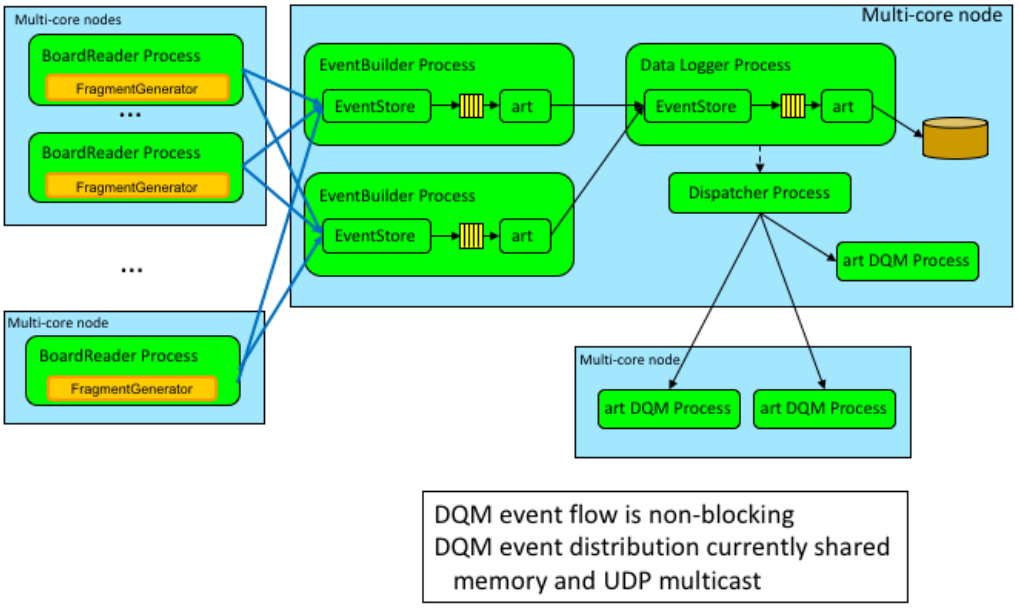

Figure 3 Diagram of a sample *artdaq* system with a focus on the ability to run stand-alone *art* processes for real-time data quality monitoring.

Figure 4 illustrates the different possibilities for data logging. Since the end of the 35-ton data taking, we have demonstrated the ability to log data from multiple Aggregators or EventBuilders. This will allow us to make use of several independent disk systems in the protoDUNE DAQ and thereby achieve the required data logging rate. In practice, either

multiple Aggregators, or multiple EventBuilders would be configured to act as Data Loggers, not both as shown in the figure – that is simply an illustration. These parallel disk-writing options will require some minor enhancements to the way that we generate system configurations, and these are well understood.

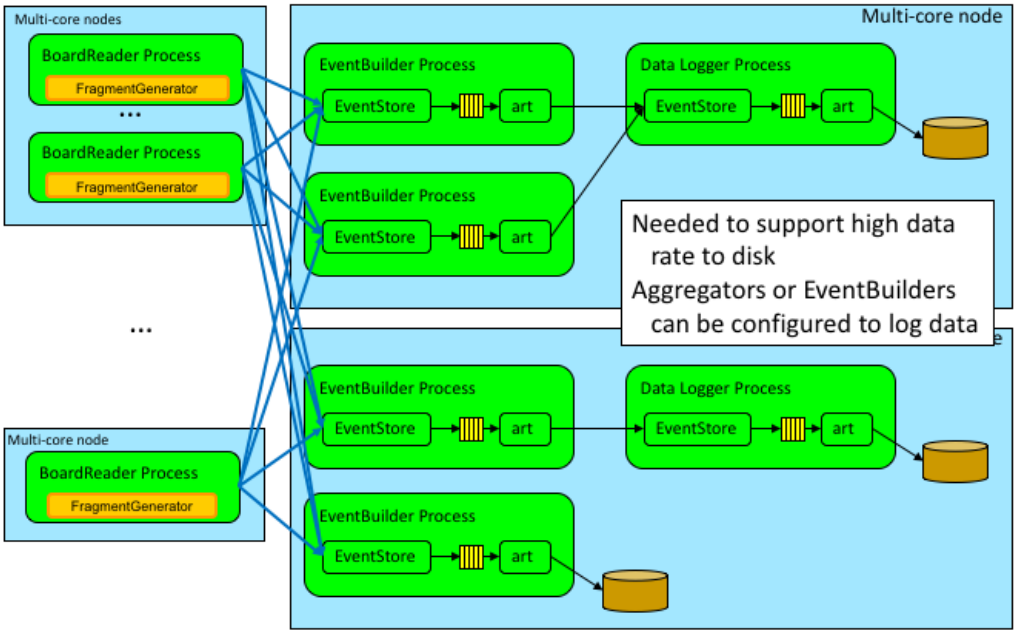

Figure 4 *artdaq* process diagram that illustrates the possibilities that exist for logging data in parallel.

Figure 5 illustrates the plan for switching to a dataflow model that is primary based on data requests (a "pull" model) for the protoDUNE DAQ. This is based on the need to properly synchronize the data fragments from each trigger. The implementation that is planned makes use of existing functionality in *artdaq*.

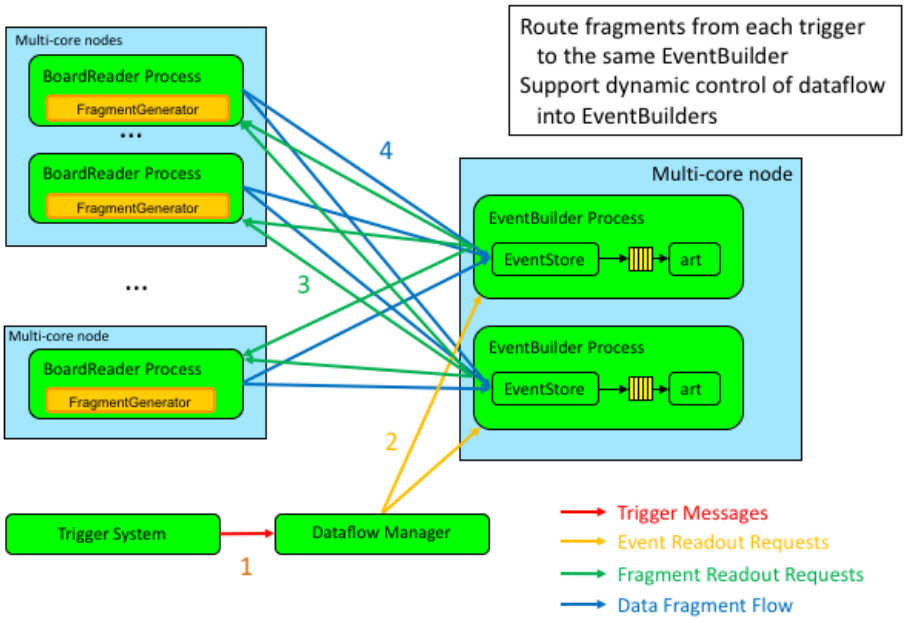

Figure 5 artdaq process diagram that illustrates the use of a Dataflow Manager process to coordinate the sending of fragments to EventBuilder instances.

Figure 6 illustrates the periodic reporting of dataflow metrics to the Trigger System BoardReader in order for it to make decisions about reducing the trigger rate when the dataflow software system is having trouble keeping up with the data volume and increasing the rate when the system has the capacity to handle that.

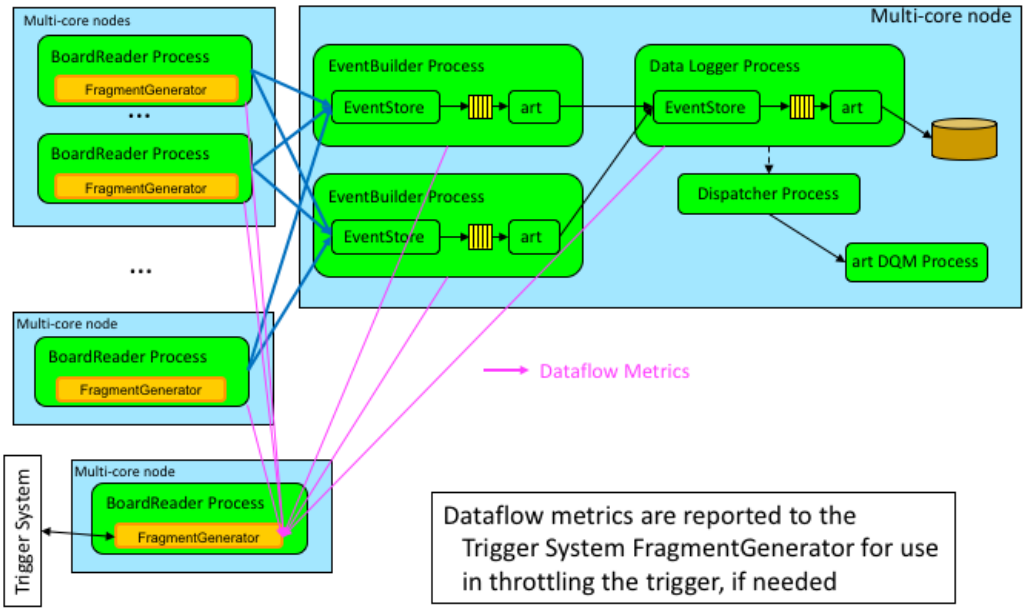

Figure 6 artdaq process diagram that illustrates the periodic sending of system performance information to the Trigger System FragmentGenerator for its use in reducing the trigger rate when needed.

Dataflow Monitoring

artdaq processes periodically report event and data rates, buffer occupancies, wait times, and other quantities to log files and various metrics-reporting tools, such as Ganglia. Additional metrics may need to be added to either the core *artdaq* processes or the FragmentGenerators

that are developed for protoDUNE, and these additions will be straightforward. There are several protocols that can be used to communicate the performance metrics to the Trigger System FragmentGenerator. We will choose one that is suitable and implement the necessary messaging.

FragmentGenerator Design

Several changes were made to the FragmentGenerator base class in artdaq recently, including the addition of several threads to handle dedicated data collection and monitoring and modifications to support a "pull" dataflow model.

In the protoDUNE DAQ, the FELIX FragmentGenerator will have the responsibility to filter the data based on notifications of beam triggers and compress the data, among the typical FragmentGenerator functions. The Trigger System FragmentGenerator will also have additional responsibilities, those of gathering trigger statistics and handling the trigger inhibit.

5 Testing

Several tests have been run to measure the performance of the *artdaq* software itself (rather than the underlying hardware). The first of these was to configure an *artdaq* system to write data to a RAM disk (which, since it consists of memory within the computer, should be very fast). In this test, a minimal set of *artdaq* processes was able to write data to the RAM disk at 1 GB/s. For reference, this sample *artdaq* system was able to handle 2 GB/s with disk writing turned off. The encouraging conclusion from this test was that the disk-writing functionality within *artdaq* will not be a limiting factor at disk-writing rates of 300-500 MB/s (which typical disk drives can handle).

The second test also used a slightly atypical set of *artdaq* processes. The goal of the test was to determine if a 10-computer DAQ cluster could handle the required 3 GB/s for protoDUNE. In this test, we used 3 computers from the Mu2e DAQ Pilot cluster and configured 3 BoardReader processes and 1 EventBuilder process on each of them. We didn't use Aggregator processes since they wouldn't have added any additional information to the test (both Aggregators and EventBuilders can be configured to perform disk writing). The goal of the test was to demonstrate 300 MB/s per node, and we used Solid State Disk drives (SSDs) to achieve this throughput since they support higher-than-typical data rates. The 300 MB/s rate per node was successfully achieved in the 3-node test, and our assertion is that this can straightforwardly be scaled up to the needed 3 GB/s by using 10 nodes. In this test, each BoardReader was configured to use a FragmentGenerator which created fake data with fragment sizes of 10 MB and fragment rates of 10 per second. Figure 7 contains a diagram that shows the layout of the test setup.

- Test environment is Mu2e DAQ Pilot cluster
- 9x3 system; BRs generating 100 MB/s (10 Hz of 10 MB fragments)
- Successfully processing 300 MB/s per node
- Caveats: MPI over Ethernet, minimizing copies, disk cache, SSD lifetime

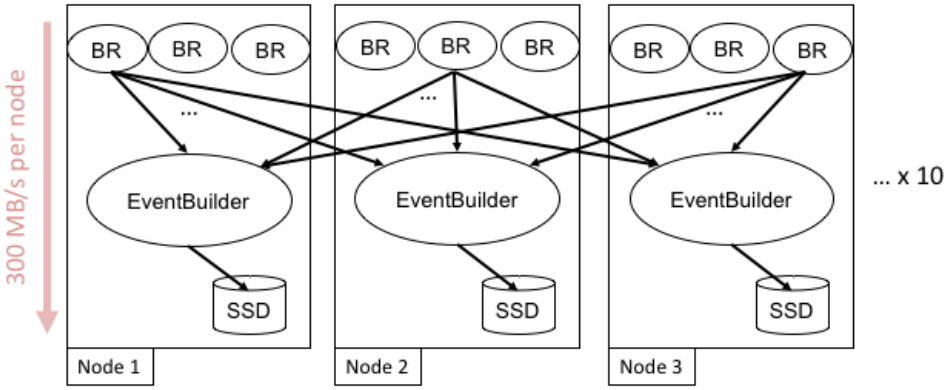

Figure 7 Description of throughput tests that demonstrated 300 MB/s per node.