Dataflow Software (*artdaq*)

Kurt Biery

protoDUNE DAQ Review

November 3-4, 2016

Overview

- Scope
- Interfaces
- Requirements
- Design
- Testing
- Risks
- Conclusion

Scope

- Configuration delivery for RCEs, FELIX modules, SSPs, trigger and timing modules, and upstream hardware
- Readout of data from RCEs, SSPs, etc.
- Event building (including synchronization of data fragments across the full detector)
- Infrastructure for software filtering, compression, or other online analysis
- Data logging
- Infrastructure for real-time data quality monitoring (DQM)
- DAQ monitoring and status message logging

- This talk does not include transfer of raw data to permanent storage

Interfaces (1)

- DAQ cluster hardware
  - Computers – number of cores, memory; processing needs
  - Networking – performance, buffering, isolation; rate requirements
  - Disk systems – read/write performance; data logging requirements
- Linux operating system
  - Network buffering, disk buffering; data transfer and logging needs
- Detector electronics
  - Configuration, readout
- Run control and configuration management system
  - Transition commands, configuration parameters

Interfaces (2)

- Trigger system
  - Communication of busy/not-busy signals; reporting of dead-time accounting and other statistics
- Real-time data quality monitoring (DQM) algorithms
  - Framework for developing, configuring, and running algorithms; interaction with developers and operators

Requirements

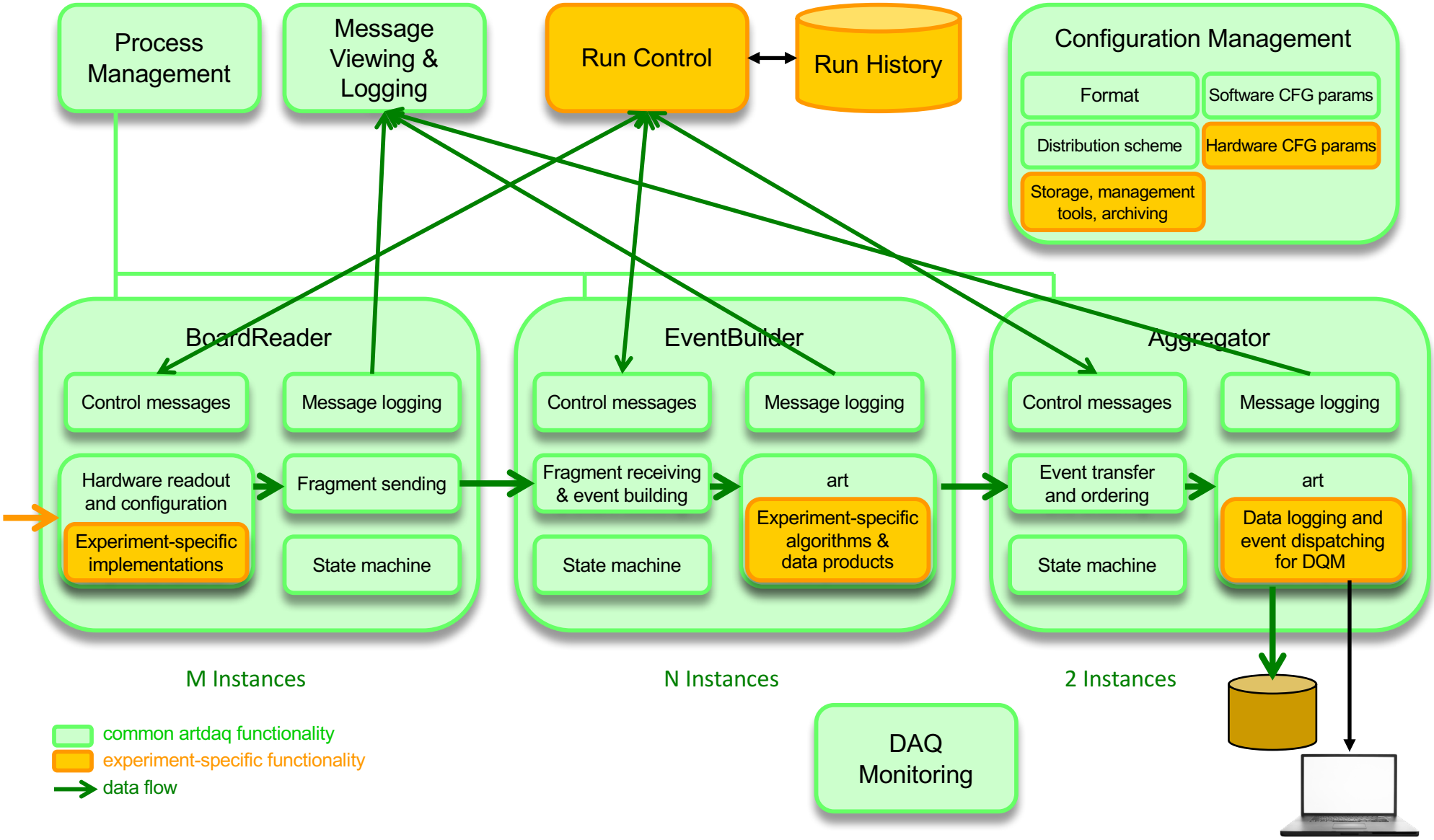
- Data throughput and logging of up to 3 GB/s.
- Support for configuration and readout of 50-60 RCEs, 24 SSPs, 2-3 FELIX cards, and trigger and timing subsystems.
- FELIX data will need to be filtered in software based on the beam trigger. Also, software compression is planned.
- Internal communication to provide trigger system busy/not-busy feedback.
- Currently, no requirement to incorporate beam instrumentation data into the TPC/PDS data path.
- More requirements listed in TDR (DocDB 1794) and earlier talks in this review.

Dataflow Software Design

Based on use of *artdaq* framework, which provides:

- Hooks for experiment-specific electronics configuration and readout (experimenters develop; common tools provided)
- Configurable numbers of readout, event building, filtering, and data logging processes
- Event processing and data quality monitoring using *art* (experimenters develop modules)
- Common functions like data transfer, event building , state model, data logging (*art*/ROOT files)
- Push and pull dataflow models supported

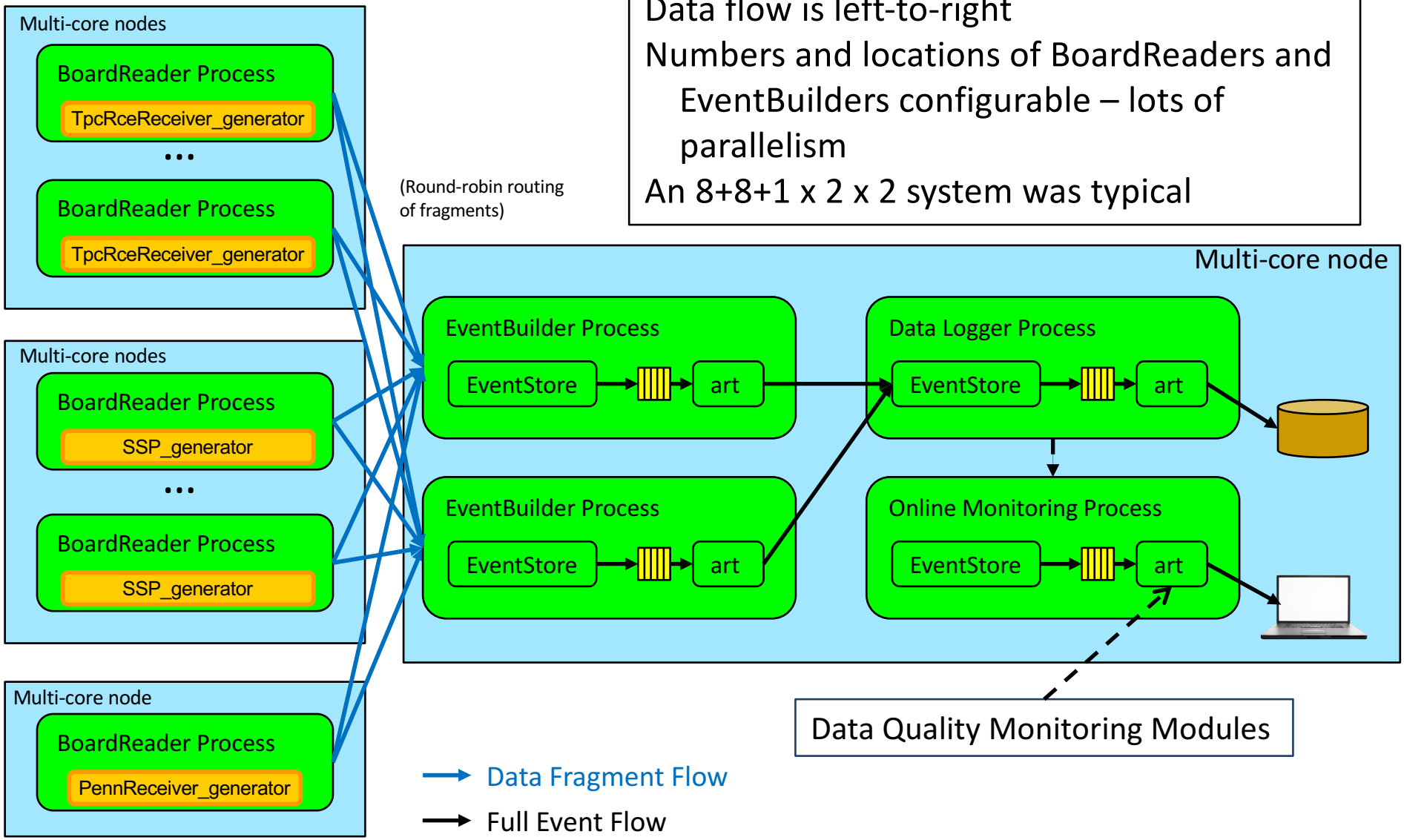
artdaq Components



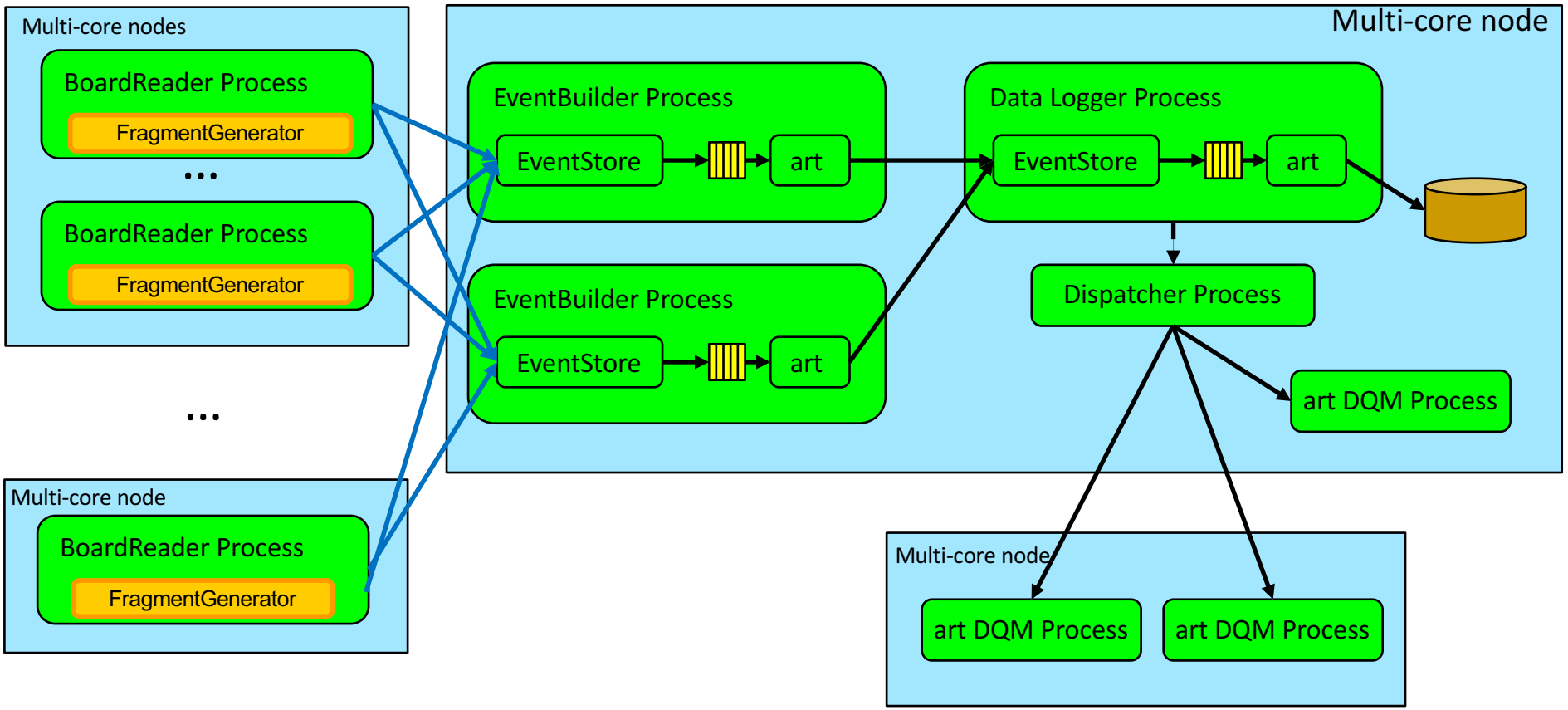
Fermilab DUNE

artdaq System for 35-ton

Data flow is left-to-right
Numbers and locations of BoardReaders and EventBuilders configurable – lots of parallelism
An 8+8+1 x 2 x 2 system was typical

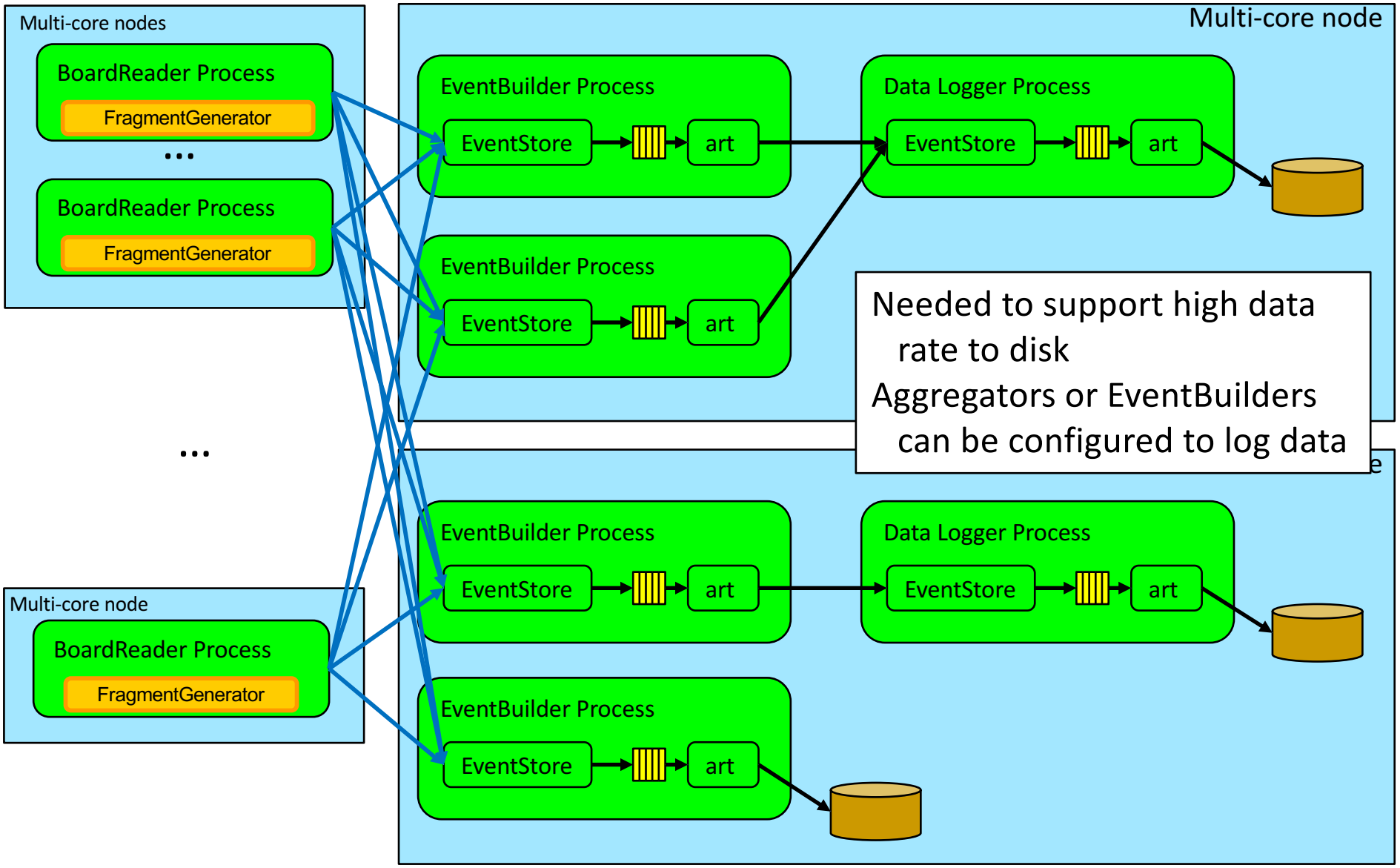


Modification A: Isolated DQM Processes

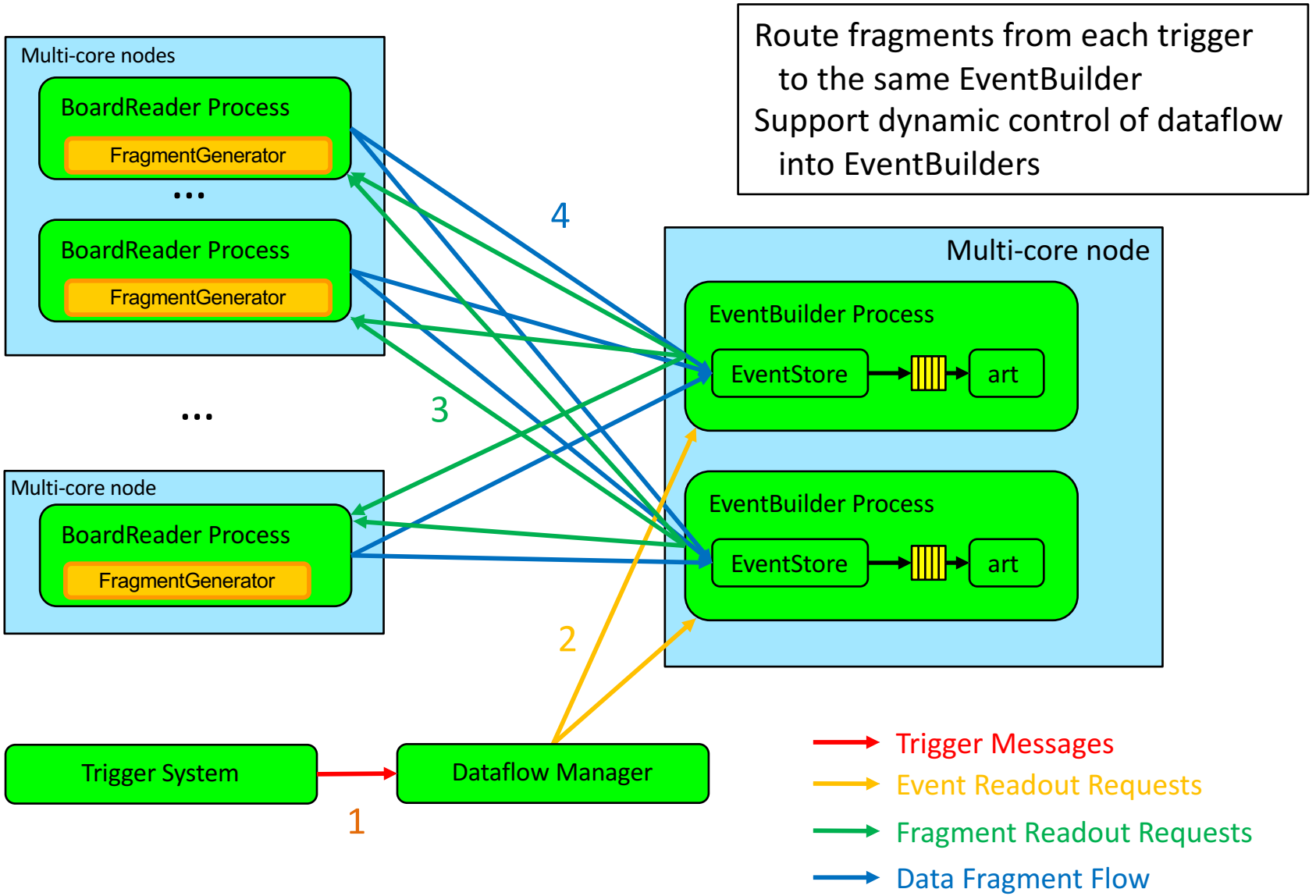


DQM event flow is non-blocking
DQM event distribution currently shared memory and UDP multicast

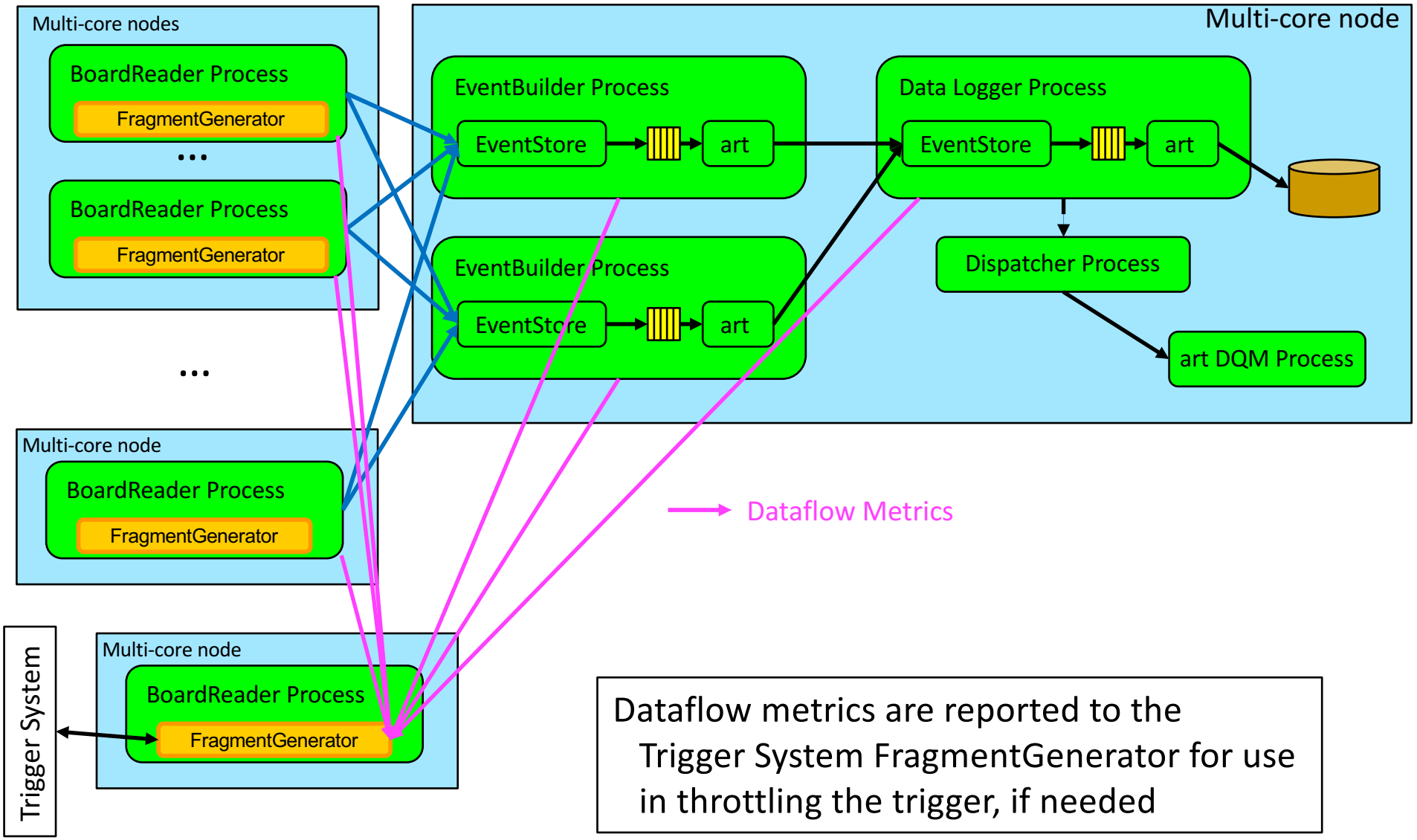
Modification B: Parallel Data Logging



Modification C: “Pull” Dataflow Model



Modification D: Feedback to Trigger System



Dataflow metrics are reported to the Trigger System FragmentGenerator for use in throttling the trigger, if needed

Dataflow Monitoring

The *artdaq* processes periodically report data&event rates, buffer occupancies, wait times, etc. to log files and Ganglia (or other monitoring packages).

- Additional metrics will be added, as needed to support the feedback to the trigger.
- A communication protocol will be chosen to communicate these to the Trigger System BoardReader.

FragmentGenerator Design

Base class enhancements since 35-ton

- Separate data-receiver thread, monitoring thread
- Modifications to support a “pull” dataflow model

Special responsibilities of the FELIX FragmentGenerator

- Filtering of fragments based on beam triggers
- Compression

Special responsibilities of the Trigger System FragmentGenerator

- Gathering of trigger statistics
- Handling of trigger inhibit

Component Testing

RootOutput_module tests (*artdaq* Aggregator and EventBuilder)

- (4x[1BRx10EB])x2AG *artdaq* system
  - No disk writing, Data Logger receives 500 MB/s
  - RAM disk writing, Data Logger writes 440 MB/s
- Single test application (*artdaqDriver*)
  - No disk writing, 2 GB/s
  - RAM disk writing, 1 GB/s

Component Testing

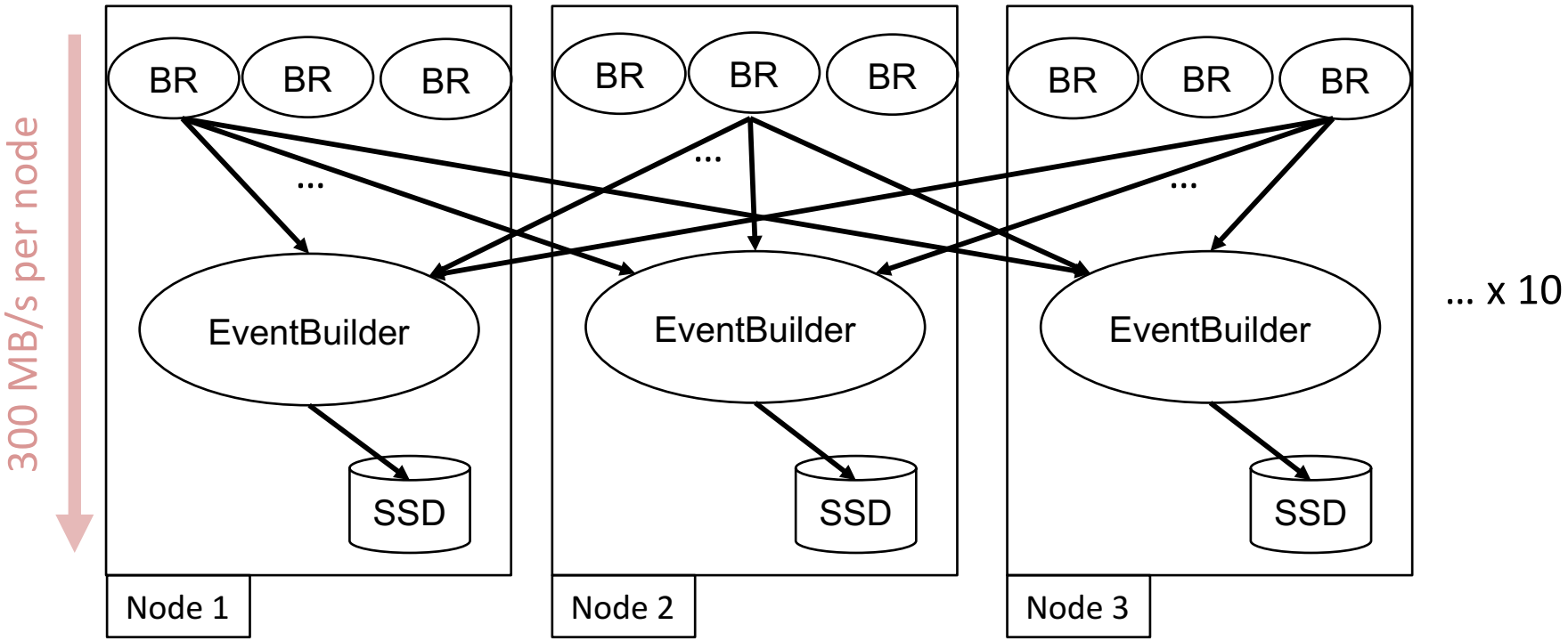
10 Gb switch, Mellanox vs. Cisco (testing tools in *artdaq*)

switch port rates:

switch	xfer_mech.	cfg	rate(MB/s)	sndrCPU%
Mellanox	MPI(Eth)	2x2	403.86	100
Mellanox	MPI (Eth)	3x3	309.74	100
Mellanox	MPI (Eth)	6x3	375.26	100
Mellanox	sockets	2x2	796.02	67.0
Mellanox	sockets	3x3	322.96	22.0
Mellanox	sockets	6x3	439.09	17.1
Cisco	MPI (Eth)	2x2	414.20	100
Cisco	MPI (Eth)	3x3	401.30	100
Cisco	MPI (Eth)	6x3	563.64	100
Cisco	sockets	2x2	860.68	77.3
Cisco	sockets	3x3	887.85	51.4
Cisco	sockets	6x3	865.80	32.6

System Testing

- 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



Future Testing

artdaq-specific:

- Typical validation of new features – ongoing
- CentOS7 – some done already, more by end of year
- Improvement in data transfer rates from MPI/Eth replacement

System tests:

- Investigate/validate DAQ cluster hardware option(s)
- Integration tests
  - Trigger inhibit messages, “pull” dataflow model
- Vertical slice tests

Lessons Learned

Some technical lessons from 35-ton:

- Important to choose appropriate cluster performance
  - (other talks on this topic)
- Linux disk cache behaviour can't be accepted blindly
  - (changes and plans in *artdaq* and Linux configuration)
- Synchronization between BoardReaders/artdaq::Fragments is important
  - Events defined by trigger in protoDUNE
- Other *artdaq* changes needed
  - (https://cdcvs.fnal.gov/redmine/projects/lbne-artdaq/wiki/Artdaq_Work_for_Summer_2016)
  - The high- and medium-priority issues have been addressed

Operational lessons from 35-ton:

- There needs to be someone responsible for testing and supporting the integrated DAQ
  - Lots of people will be present at CERN for integration

Risks

- Insufficient computing, network, or disk performance
  - Additional *artdaq* processes can be configured, within reason
- Difficulties in FragmentGenerator development
  - Assistance from *artdaq* team
- Integration issues with new features or new electronics
  - Support from *artdaq* team

Conclusion

- The dataflow software design meets the requirements for data rates and functionality.
- The changes to the core artdaq system are not numerous and can can certainly be accomplished by the time that they are needed.
- The applicable lessons-learned have been incorporated.