Discussion of the MLT-DFO interaction

Kurt Biery

23 Nov 2021

FarDet Physics Performance/Data Selection Meeting

Background

We are working on the introduction of the DFO (Dataflow Orchestrator) to our DUNE-DAQ systems.

- Recall that the DFO is planned to be the Dataflow system's interface to the Trigger system, and the DFO is planned to distribute TriggerDecisions to available TriggerRecordBuilders (event builders) within Dataflow.

In v2.8.2 and earlier software, the ModuleLevelTrigger (MLT) interacted directly with Dataflow modules.

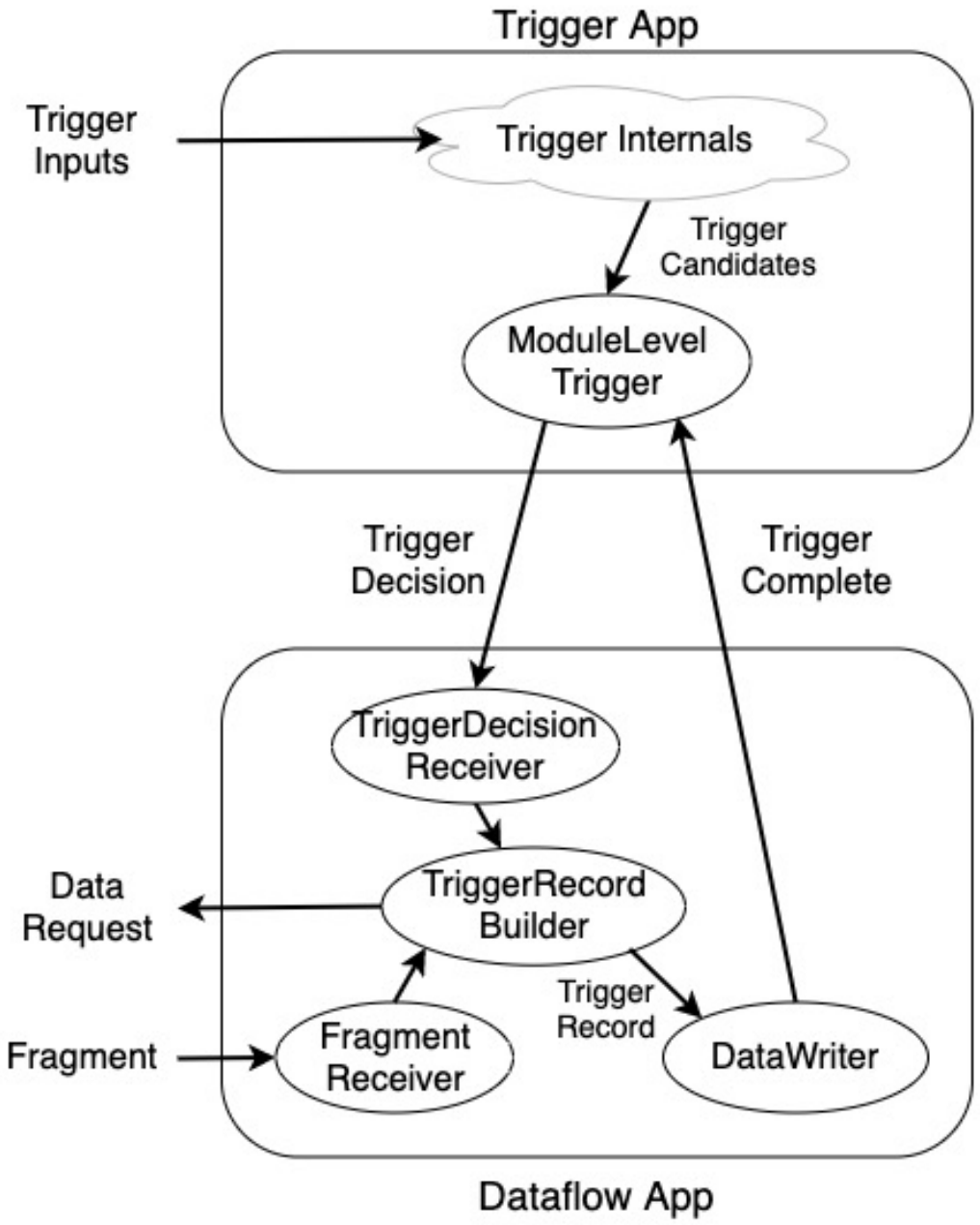
In addition, the throttling of TriggerDecisions (TDs) has been handled by the MLT based on a config parameter that tells it the maximum number of TDs the Dataflow system is willing to handle at any one time.

Underlying assumption

In all of what I'm presenting, there is the assumption that there may be times that the Dataflow system will be unable to process yet-one-more-trigger (because all of its buffers/resources are in use).

My assumption is that the Trigger/MLT will drop triggers that happen when such a condition is present.

Of course, I/we expect that the Trigger system will suitably account for such losses.



Notes on current system:

- MLT interacts with TDReceiver and DataWriter
  - This is straightforward in the current system, with only one of each, but more in the future
- MLT handles trigger throttling based on one of its config params

Features of what we have now

A Trigger system entity…

- Handles the internal calculations are needed to determine the components of the {trigger livetime/deadtime/whatever} calculation that are independent of unavailable buffers within Dataflow
- Knows directly when a trigger decision needs to be dropped on the floor because the Dataflow system does not have an available buffer to handle the resulting TriggerRecord
- Also knows when TriggerRecords are successfully written to disk
  - However, the “TriggerComplete” message is very terse (just the run and trigger numbers, no information about missing Fragments, etc.) and this message is not currently full-featured/robust.

Proposal

Move the tracking of free DF buffers and assignment of TDs to TRBs to the DFO.

- In the spirit of having the DF system handle its internal details
  - Better encapsulation – Trigger doesn't need to talk to individual DF modules
- Likely uncontroversial, but still a change (Trigger system needs to be told about lack of buffers instead of knowing it directly)
- Trigger throttling would still be handled by the MLT, of course, but it would be based on information provided by the DFO

If we can agree on this, we still need to agree on the specific MLT-DFO interaction.

Additional aspect: remove (from Trigger) the knowledge that the TR was successfully written to disk.

Cost/benefit of what is being proposed

Benefit: all of the gory details of buffer management within Dataflow system (e.g. software development, maintenance) is handled by the Dataflow subsystem and Working Group.

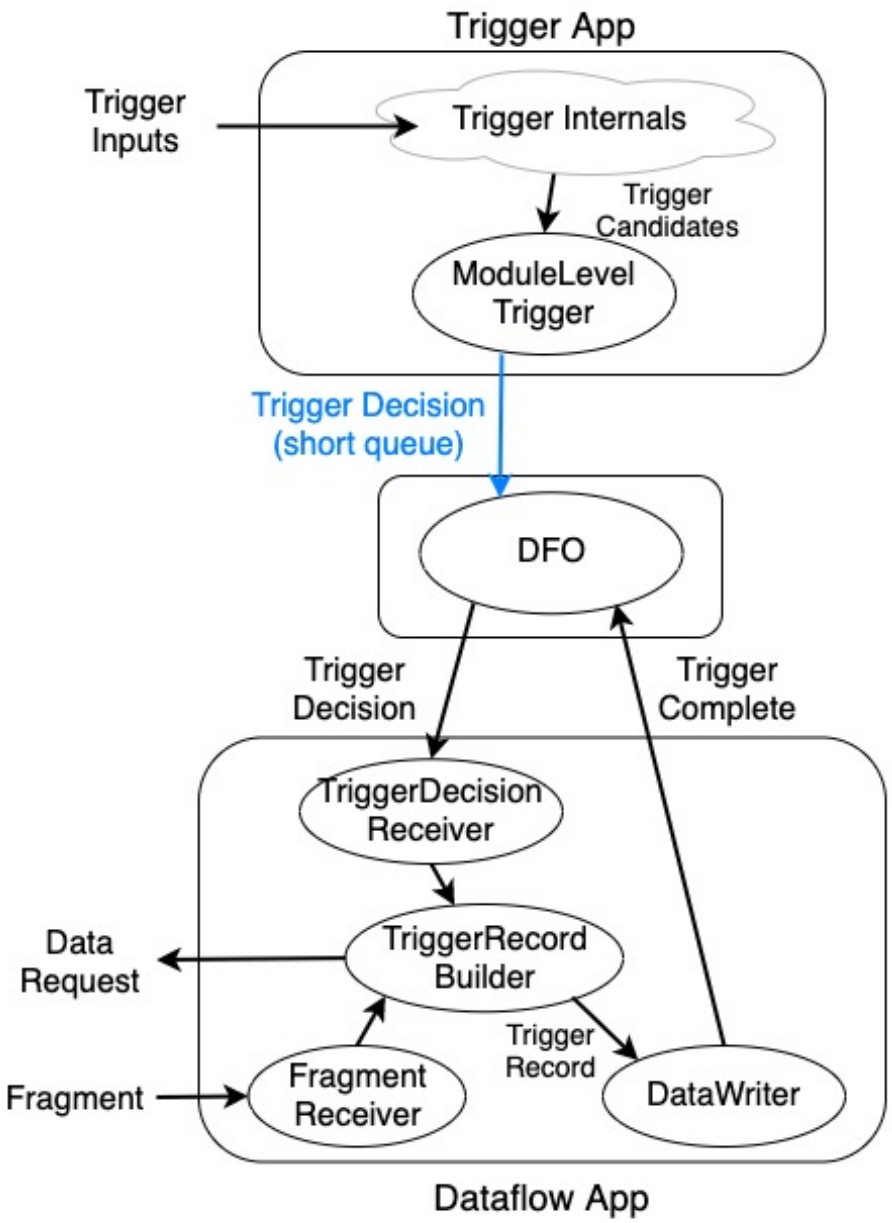
Cost: the Trigger system does not have direct knowledge of the number of buffers that are available and in-use.

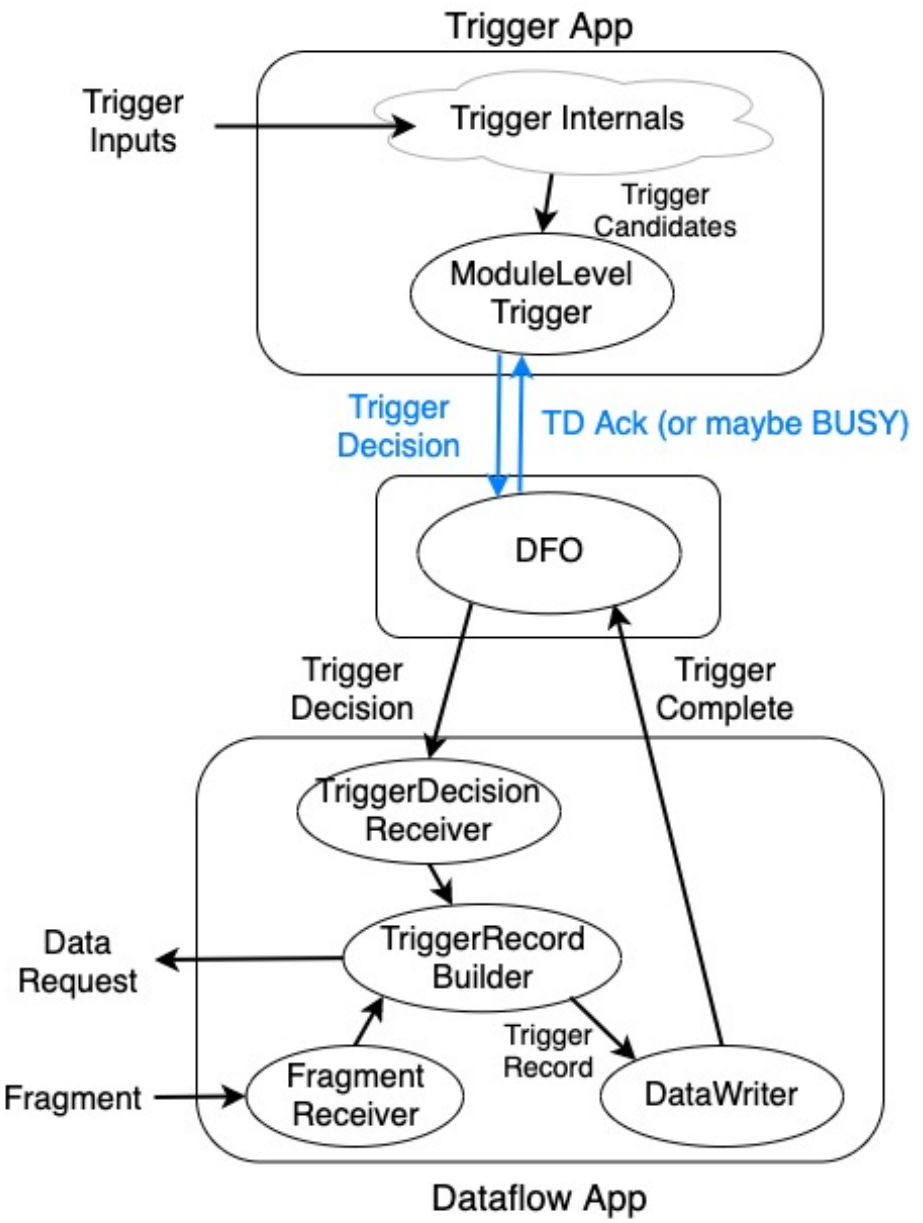
Features of what is being proposed

A Trigger system entity…

- Handles the internal calculations are needed to determine the components of the {trigger livetime/deadtime/whatever} calculation that are independent of unavailable buffers within Dataflow
- Knows very quickly when a trigger decision needs to be dropped on the floor because the Dataflow system does not have an available buffer to handle the resulting TriggerRecord
- No longer would know when TriggerRecords are successfully written to disk

A couple of options for the MLT-DFO interaction





Some details on the proposals

A detail about the short-queue model:

- The idea is that the MLT would check if there is space on the Queue to push a TD, and if not, would drop the trigger

A detail about the ‘Ack’ model:

- DFO immediately returns TD Acknowledgement indicating its acceptance or rejection of a commitment to fully process the resulting TriggerRecord

In both of the proposed models, the tracking of any problems in the dataflow is the responsibility of the Dataflow system.