

artdaq - Bug #23426

RootDAQOut determines filename before rank and app_name set in art v3

10/15/2019 11:09 AM - Eric Flumerfelt

Status:	Closed	Start date:	10/15/2019
Priority:	High	Due date:	
Assignee:	Eric Flumerfelt	% Done:	100%
Category:		Estimated time:	0.00 hour
Target version:	artdaq v3_07_00	Co-Assignees:	
Experiment:	DUNE		
Description			
Kurt has seen issues at protoDUNE related to RootDAQOut not having the correct app_name or my_rank values when the module is being constructed in art v3 (s83). I have seen similar issues with the demo, where mediumsysteem_with_routing_master produces output files with names like "artdaqdemo_r000022_sr01_2_dl-13_dl1.root".			

History

#1 - 10/15/2019 01:52 PM - Kurt Biery

As a work-around at protoDUNE, I've copied the code that determines the app_name and my_rank from SharedMemoryReader (which is used in the EventBuilder InputSource) to RootDAQOut-s81/RootDAQOut_module.cc. With this, the system runs for more than a few events at protoDUNE. (Before this change, it would only run for ~10 events with one EventBuilder.)

#2 - 10/16/2019 10:39 AM - Eric Flumerfelt

- Assignee set to Eric Flumerfelt

- Status changed from New to Assigned

I'm working on a solution involving creating a new service that handles all interactions with the artdaq shared memory. By doing this, we can make sure that my_rank and app_name are set by getting a handle to the service before attempting to use those variables.

Implementation will be on:

artdaq-core:bugfix/23426_SMER_IsEndOfDataFunction
artdaq:bugfix/23426_ArtdaqSharedMemoryService

#3 - 10/17/2019 09:26 AM - Eric Flumerfelt

- Status changed from Assigned to Resolved

I have tested these changes by running the simple_test_configs (./run_integration_tests.sh;./run_integration_tests.sh --runs 3 --runduration 60) in both s82-e19-prof and s67-e17-prof, and verified that all data files were created with appropriate names, and contained the expected number of events with the expected data.

#4 - 11/04/2019 05:34 PM - John Freeman

- % Done changed from 0 to 100

I've performed a pair of runs using an s83-e17 installation of artdaq-demo, described in mu2edaq01:/home/jcfree/run_records/3093 and /home/jcfree/run_records/3096. In both cases, I ran with a datalogger on mu2edaq01 and a datalogger on mu2edaq11, both writing to /tmp. In run 3093, I ran using the develop branches of artdaq-core and artdaq, as well as the develop branch of DAQInterface. In run 3096, I ran with this issue's feature branches for artdaq-core and artdaq; additionally, I modified the configuration so it used the ArtdaqSharedMemoryService. In run 3093, I produced these two root files:

```
mu2edaq01:/tmp/artdaqdemo_r003093_sr01_1_dl10.root  
mu2edaq11:/tmp/artdaqdemo_r003093_sr01_1_dl10.root
```

Whereas in run 3096, I produced these two:

```
mu2edaq01:/tmp/artdaqdemo_r003096_sr01_1_dl14.root  
mu2edaq11:/tmp/artdaqdemo_r003096_sr01_1_dl15.root
```

For both runs, the datalogger on mu2edaq01 was rank 3, and on mu2edaq11 it was rank 4, so it appears the feature branches give us the result we want.

#5 - 11/04/2019 05:35 PM - John Freeman

- Status changed from Resolved to Reviewed

#6 - 11/12/2019 11:17 AM - Kurt Biery

In looking at the code changes associated with this issue, I've come up with two questions so far...

1. In SharedMemoryReader::readNext() my sense is that the older implementation allowed for a timeout that was (gracefully?) interpreted as a shutdown request, whereas with the newer implementation, there seems to be the possibility of looping indefinitely calling shm->ReceiveEvent(false). Is there a way that the loop of ReceiveEvent() calls can be exited in the newer code that I haven't noticed yet? Or, is there some other mechanism by which that loop can be stopped?
1. One of the changes seems to have been the collapsing of the receiveEvent and receiveInitMessage methods (in NetmonTransportService) with the ReceiveEvent(boolean broadcast) method in ArtdaqSharedMemoryService. I haven't fully understood all of the ramifications, but this seems like a useful change. The question is whether there are broadcast messages that we can specifically test, or watch for in our testing, to validate the new code. Are there other broadcasts besides Init messages? Are there system layouts that will better exercise the broadcasts than others?

#7 - 11/12/2019 12:11 PM - Eric Flumerfelt

One feature of SharedMemoryEventReceiver is that it will always prioritize broadcasts over data so that they always get handled in a timely fashion. Broadcasts are currently limited to Init, EndOfRun, and EndOfData.

Both NetMonWrapper and SharedMemoryReader have that while loop around the ArtdaqSharedMemoryService call, which in turn has its own similar while loop. I think we could definitely improve the handling there, at the very least adding some warning-level TRACES. I'm not sure about introducing timeouts, as some experiments have had issues with very low event rates breaking things...(the default timeout was one day).

#8 - 11/12/2019 01:48 PM - Kurt Biery

Thanks for the info.

I ran some tests in which I put DAQInterface in 'direct process control' mode and then artificially prevented any of the BoardReaders from sending data to the EventBuilders. The EventBuilders (and their associated art processes) still shut down cleanly. I presume that this is because the EndOfData fragment (broadcast) gets through in both cases and exits the 'forever' loop that I was concerned about. Given that, and given the fact that EventBuilders can (and will) forcibly shut down art processes that take too long to gracefully shut down, my sense is that the new code is fine as it is. And, if I understand your point about the previous timeout being set to one day, then the new code is effectively not much different than the old code (so we shouldn't see different behavior).

#9 - 11/14/2019 10:29 AM - Kurt Biery

I have merged the changes in the three repositories (artdaq, artdaq-core, and artdaq-utilities-daqinterface) into the respective "develop" branches and updated the 'artdaq branches' wiki page.

#10 - 11/21/2019 02:37 PM - Eric Flumerfelt

- Target version set to artdaq_core v3_05_09

- Status changed from Reviewed to Closed

#11 - 11/21/2019 02:46 PM - Eric Flumerfelt

- Target version changed from artdaq_core v3_05_09 to artdaq v3_07_00