ds50daq - Bug #4789

Why is MonitorSer fitting histograms in the middle of a run?

10/11/2013 09:16 PM - Kurt Biery

Status:	Assigned	**Start date:**	10/11/2013
Priority:	Normal	**Due date:**	10/18/2013
Assignee:	Michael Wang	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:	NeededEnhancements		

Description

Alessandro reported:

- I see in the log from time to time that the MonitorSer is trying to fit spectra in the middle of a run

My observation:

- When I was doing my file-closing tests several days ago, **and** I had the online monitoring running, I noticed that the histograms would reset every time a file was closed and a new one opened. I think that this is the same effect as what Alessandro is seeing. At the time, I thought that it was just an unfortunate side-effect, but now I'm not sure why it is happening. We don't send an endRun event at pause/resume, just an endSubRun...

Please investigate what is happening here. The online monitoring modules take various actions at beginRun and endRun times, but it is not clear why those methods would be called at DAQ pause and resume times.

This problem will be easily reproduced at the WH14 testatand, and I can demonstrate it whenever it is convenient.

Related issues:

Related to artdaq - Idea #5961: Coordinate the requested changes to art for a...	**Closed**	**04/18/2014**

History

#1 - 10/14/2013 11:49 AM - Kurt Biery

Mike,
Here are the steps to reproduce this at the WH14 teststand:

- update your local ds50daq git repository on dsfr6 to have the latest code on the develop branch; build the ds50daq software
- in one shell window, setup the ds50daq development environment, and run 'startWH14Test1System.sh'
- in a second shell window, setup the ds50daq development environment, and run
  - 'manageWH14Test1System.sh -p 1000 -s 2000 -m on init'
  - 'manageWH14Test1System.sh -N <runNumber> start'
  - 'manageWH14Test1System.sh stop'
- in a third shell window on dsfr6, do **not** setup the ds50daq environment, but run the following commands:
  - 'xpra start --no-pulseaudio --enable-sharing :50' # may not be needed if someone else has already started the server
  - 'xpra attach ssh:dsfr6:50 --enable-sharing'

After the file(s) in /data/test reach ~2 GB in size, they will automatically be closed, and you'll see the online monitors reset. And, the effect may be more dramatic - in my test today, I saw the MonitorSer module crash, probably because it was trying to fit an empty histogram.

In any case, it is the underlying problem that we're curious about - why is endRun (or beginRun) getting called when the system pause/resumes.
Thanks,
Kurt

#2 - 10/14/2013 12:33 PM - Kurt Biery

I forgot to include the patch to WFViewer_module.cc

[biery@dsfr6 ArtModules]$ git diff WFViewer_module.cc
diff --git a/online/ArtModules/WFViewer_module.cc b/online/ArtModules/WFViewer_module.cc
index 54e3285..ac49bf9 100644
--- a/online/ArtModules/WFViewer_module.cc
+++ b/online/ArtModules/WFViewer_module.cc
@ -149,15 +149,15 @ void ds50::WFViewer::beginRun(art::Run const &) {
canvas_[2]~~-> Update();~~
~~((TRootCanvas*)canvas_[2]~~->GetCanvasImp ())->DontCallClose ();

- try {

```
-    art::ServiceHandle<ds50::DBInterface> dbi;
-    ds50::db::result r = dbi->latest ("calib.pmt_geometry", "");
-    for (size_t i = 0; i < r.cell_elements("r", 0); i++) normals_.push_back (r.get<float>("r", 0, i) >= 0);
- } catch (const std::runtime_error& e) {
-    mf::LogError ("WFViewer") << "cannot read db, proceeding with default: " << e.what ();
+  //try {
+  //  art::ServiceHandle<ds50::DBInterface> dbi;
+  //  ds50::db::result r = dbi->latest ("calib.pmt_geometry", "");
+  //  for (size_t i = 0; i < r.cell_elements("r", 0); i++) normals_.push_back (r.get<float>("r", 0, i) >= 0);
+  //} catch (const std::runtime_error& e) {
+  //mf::LogError ("WFViewer") << "cannot read db, proceeding with default: " << e.what ();
normals_.resize (40, true);
normals_[38] = normals_[39] = false;
- }
+    //}
}
```

#3 - 10/14/2013 02:23 PM - Kurt Biery

- Status changed from New to Assigned

The code that automatically closes/opens files is in ds50daq/DAQ/Aggregator.cc. Please look for the use of, and definition of, the sendPauseAndResume_() method.

This reminds me that, in fact, I believe that we can trigger this odd behavior simply by sending Pause and Resume commands to the system with manageWH14Test1System.sh.

#4 - 10/17/2013 01:39 PM - Michael Wang

-------- Original Message --------
Subject: art behavior in an end of subrun
Date: Wed, 16 Oct 2013 15:51:16 -0500
From: Michael Wang <mwang@fnal.gov>
To: Kurt Biery <biery@fnal.gov>

Hi Kurt,

As you had explained, the way you do auto-file-closing is to send a pause command which forces the current subrun to end and the output files to be closed.

As you know, when we send a pause, we generate end-of-subrun fragments. When RawEventQueueReader detects these fragments, it returns an event with valid Run number but with "flush" subrun and event numbers (this has always been the behavior with non-empty runs/subruns even prior to the changes we introduced to fix empty runs/subruns). The new flush subrun and event numbers cause Source.h to first send a "SubRun" event to the state machine followed later by an "Event" event. These events trigger a series of actions in the state machine ending up in the HandleEvents state.

When the RawEventQueueReader detected the end-of-subrun event, one other thing it did was to set the outputFileCloseNeeded flag to true. This causes a subsequent call of RawEventQueueReader to return false. This false causes a "File" event to be generated and sent to the state machine. The incoming "File" event causes a transition of the machine from the HandleEvents state to the HandleFiles state. In leaving the HandleEvents state in order to transition to the HandleFiles state, not only the exit actions of HandleEvents need to be called, but even those of all outer states from which it is derived, up to but excluding the innermost one that is in common with the state we are transitioning to. In other words, the exit actions of both HandleSubRuns and HandleRuns need to be called before transitioning to the HandleFiles state.

So it is my understanding that the exit actions of HandleRuns triggers the endRun methods of the output modules to be called when a pause is sent. I think this is what is causing MonitorSer to fit the histograms during an auto-file-closing when you use the pause command to implement this feature.

Mike

#5 - 10/19/2013 07:25 AM - Michael Wang

-------- Original Message --------

Subject: Re: Fwd: art behavior in an end of subrun
Date: Fri, 18 Oct 2013 09:48:26 -0500
From: Chris Green <greenc@fnal.gov>
To: Michael Wang <mwang@fnal.gov>
CC: Kurt Biery <biery@fnal.gov>, artists <artists@fnal.gov>

On 10/16/13 4:00 PM, Michael Wang wrote:

> Hi Chris,
>
> Kurt asked me to look into a ds50daq issue [redmine bug #4789] where he sees some odd behavior in ds50daq MonitorSer output module after implementing his auto-file-closing feature. He implemented this feature by sending "pause" commands that end a subrun and trigger the ouput file closing. But he noticed that this endSubRun would also trigger an endRun/beginRun in his ouput Module causing it to do certain things (fit histograms) meant to be done only at a real end-of-run.
>
> I looked into this a bit and below is my description of what I think is going on. I just want to remind you again that I am no art expert and my understanding can be totally wrong -- so please let me know if this is the case. Kindly let us know if this might explain what is going on and if you can perhaps propose a solution.
>
> Let us know when you might be able to meet with Kurt and me briefly to discuss this issue. I think Kurt said 9-11 tomorrow is not a good time for him.

Hi Mike,

Sorry for the delay getting back to you on this. We have had several short deadline emergencies this week including the discovery of a possible misbehavior in DS50 compression / decompression which needs to be understood quickly for wider implications.

I believe that your description of the behavior below is accurate. My first question is, why did you write an output module (MonitorSer) rather than an analyzer? An output module is significantly harder to write, and based on your (admittedly brief) description I'm not sure why you would need that would buy you anything.

Unfortunately, the only thing I can think of that would solve your problem is the implementation of the long-anticipated re-imagining of the run and subrun system to include run and subrun, "fragments" and a separate, non-payload-carrying end-{run,subrun} signal. One would then put the histogram fitting into the response to the EOR signal rather than the subrun-fragment routine. This may or may not need to be combined with the state machine overhaul which would change the end-of-file handling to be pull rather than its current push.

If you would like to discuss this further, I should be able to meet with you both Monday, except 11:30-noon and 2-3pm.

Thanks,
Chris.

#6 - 04/23/2014 01:22 PM - Kurt Biery

- Target version set to NeededEnhancements