

## artdaq - Idea #14661

### Can artdaq::Fragment have zero dependencies on other code in the art stack?

11/23/2016 03:00 PM - Kurt Biery

<b>Status:</b>	Closed	<b>Start date:</b>	11/23/2016
<b>Priority:</b>	Normal	<b>Due date:</b>	
<b>Assignee:</b>		<b>% Done:</b>	0%
<b>Category:</b>	Under Consideration	<b>Estimated time:</b>	0.00 hour
<b>Target version:</b>	artdaq v2_02_00		
<b>Experiment:</b>	-		

#### Description

Recently, a couple of folks on protoDUNE wondered whether it might be possible to look at raw data files written by *artdaq* with bare ROOT tools. The stated purpose for this way of looking at raw data would be for quick debugging. It was suggested that having a way to build some fraction of the data-access classes outside of cetbuildtools might be helpful in this regard.

One first step in this direction would be to minimize the dependencies of artdaq::Fragment on other code in the *artdaq/\_art\_stack*. This would probably entail moving it into a package that has very limited dependencies.

Ron may already be looking into this.

#### History

##### #1 - 11/23/2016 03:02 PM - Kurt Biery

- Subject changed from *Can artdaq::Fragment have zero dependencies on other code in the \_art\_stack?* to *Can artdaq::Fragment have zero dependencies on other code in the art stack?*

##### #2 - 01/13/2017 03:49 PM - Eric Flumerfelt

- Category set to *Under Consideration*

- Target version deleted (577)

We're pretty sure at this point that there's minimal reduction possible here, as an "artdaq-data" product would still have to rely on cetlib (for BasicPlugin), and that drags boost along with it (which means that we would still be requiring 2/3 of the mass of the product stack (gcc comes, too)) to build the thing.

##### #3 - 03/02/2017 10:40 AM - Eric Flumerfelt

- Status changed from *New* to *Closed*

- Target version set to *artdaq v2\_02\_00*

*artdaq\_core* now depends on *canvas*, which is as low as we can practically go. If someone wants to deal with Fragments in an entirely binary fashion, then we should consider that issue separately. There also still remains the possibility of creating a version of BoardReader with minimal dependencies for running on embedded systems.