

Run Control, Operational Monitoring, and Configuration

Wesley Ketchum

ProtoDUNE DAQ Design Review

November 3 2016

Introduction

- Run control
 - Suite of software managing the state of DAQ
 - Interface between shifters/operators and DAQ
- Operational monitoring
 - Monitoring of the state of DAQ processes and data flow to ensure smooth operation of the DAQ
 - DAQ process monitoring (what is running? in what state?)
 - DAQ node monitoring (are the nodes healthy? disk usage? cpu load?)
 - DAQ data flow monitoring (how much data from each node? buffer occupancies?)
- Configuration
 - Storage of DAQ configuration for online and offline use
 - Interfaces for updating/creating new configurations
 - Interfaces to run control for choosing and passing configurations to DAQ components

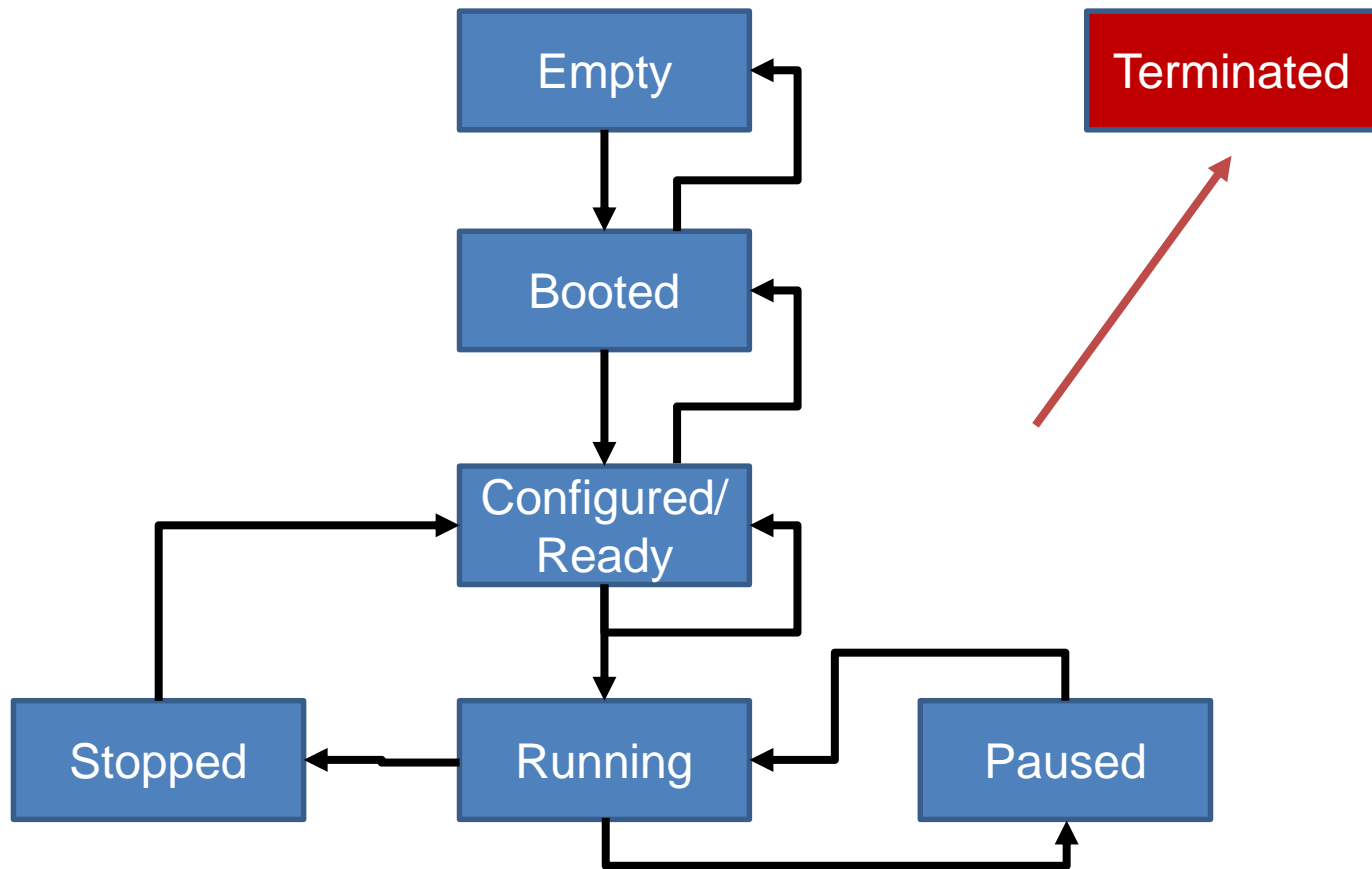
Quick detour: what is a run?

- We define a run as a period of contiguous data-taking from detector components in a static configuration state
 - Change of detector configuration → change of run
 - A run history database will provide basic information on runs taken
 - Start/end times, configuration ID/name used, run type, etc
 - Detector state/conditions can change during run
 - Operational monitoring / slow monitoring
- An event is the total collected data from a contiguous period of time during an individual run
 - For ProtoDUNE, usually defined by some triggered readout
 - Each event has a unique identifier
 - Run number and event number (with optional subrun number)
 - Each event has a unique “start” or “triggered” time

Requirements

- Provide clean interface between shifter control and existing DAQ control infrastructure
 - Integrated run control not currently provided with *artdaq*, but procedure for process management and control message passing exists
- Integrate DAQ operational monitoring with monitoring of other hardware components
 - Centralized monitoring → simpler operation and easier debugging
- Ability to alarm on DAQ status and internal data flow status
 - Interfaces for customized DAQ monitoring metrics
- Easy to maintain and modify configurations
 - Flexibility for new configuration parameters
- Configuration parameters fully accessible for offline reconstruction
 - And maintained over expected lifetime of data

artdaq State Machine



artdaq Control Processes

- Processes started using custom/simple Process Management Tool (PMT)
 - Currently started via MPI as MPI ranks used for data transfer pathway
 - Upcoming modifications: socket-based data transfer
 - More flexibility in how to start processes in PMT
- XML-RPC commands used for communication to DAQ processes
 - Check status, and pass through state transitions
- Experiment-specific long-running interface programs have previously been used for calling these processes
 - e.g. DAQInterface in 35-ton

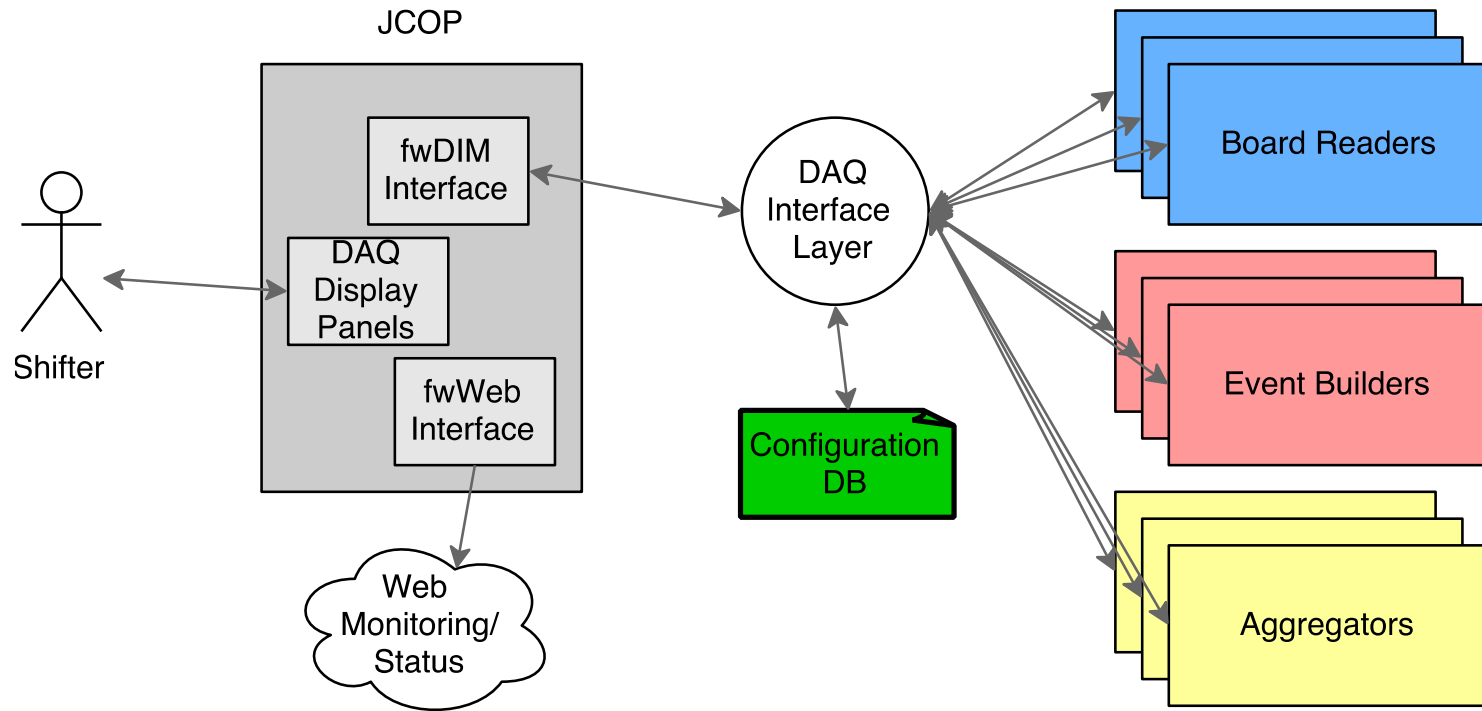
Run control: JCOP

- JCOP: Joint Controls Project
 - Grouping of representatives from CERN experiments
 - Developed framework of common software components for control systems
 - Hardware and software interfaces
 - Uses WinCC-OA for supervisory control and data acquisition
 - Control interfaces, custom data point elements, alarm capabilities, storage/archiving of controls data, and single event manager server subsystem managers
- Well-developed and common system at CERN
 - Benefit from expertise/common tools from many experiments
 - CERN support and expertise for JCOP system
- ***Decision to integrate DAQ into this system***
 - Where the significant work now is ongoing

Interface of DAQ to JCOP

- Maintain existing *artdaq* infrastructure, but build interface layer to JCOP
- Consulting experts → use CERN's DIM (Distributed Information Management System) as interface protocol
 - Simple server/client model
 - Python and C++ APIs
 - Interface from DIM to JCOP already developed (fwDIM)
- Interface plan
 - Write parent control with DIM server
 - Write DIM client to issue commands → PMT/XMLRPC
 - Write DIM client to receive back DAQ state
 - Integrate DIM client into JCOP
 - Developing DAQ process data elements with necessary state

Block diagram



Operational monitoring in JCOP

- JCOP also good choice for operational monitoring
 - Integrated controls/alarming/archiving of status with other components
- *artdaq* provides simple metric manager plugins to allow display of built-in and custom monitoring metrics
 - Monitoring metrics easily defined in DAQ software code
 - Plugins exist for ganglia, graphite, text files, EPICS
 - Can run multiple monitoring plugins at same time
- Interface plan
 - Both DIM and WinCC-OA have C++ APIs, so plugins can exist for both
 - Choice of DIM would imply then interfacing DIM servers to JCOP
 - Likely start with plugin for DIM server (looks more fault-tolerant on *artdaq* side)
 - Then create data point elements in JCOP for describing DAQ monitoring metrics

JCOP at NA62

Trigger Flow

L0 **1532117** from LOTP Red. Fact.

L1 in **1513109** out **206691** of which SP AP

L1 data **206691** REQUESTED Red. Fact.

L2 in **205336** out **205336** of which SP

Merger in **205336** AP

out **205337**

- Primitives
- Calibration
- Synchro
- Control
- Periodic1
- Periodic2

Run Infos

Run Type

Start Time

End Time

Beam Type

Shift crew

StartRun

Comment

EndRun

Comment

RunNumber Burst #

Burst State

Primitives Count

CHOD	2.23e+07
RICH	1.83e+07
LAV	3.13e+06
MUV	2.10e+07
NCHOD	2.14e+07
TALK	0.00e+00
LKr	3.64e+06

Triggers

Control	2.21e+07	400
MASK0	3.18e+06	200
MASK1	8.91e+05	1
MASK2	2.06e+05	1
MASK3	5.66e+05	5
MASK4	7.57e+05	2
MASK5	1.74e+06	50
MASK6	1.12e+05	1

Exp. scalers

QX	2.75e+06
Q1-OR	9.58e+06
MUV1 OR MUV2	0.00e+00
MUV3	1.06e+07
NHOD	5.20e+05
IRC	2.70e+06
CHANTI	4.87e+06
ECN3_008	0.00e+00
ECN3_009	0.00e+00
ECN3_010	0.00e+00
ECN3_011	0.00e+00
ECN3_012	0.00e+00
ARGONION	5.28e+08

Beam Infos

Page1 comment

T10 Intensity [e¹¹]

T10 Symmetry

Merger

	Proc. Burst	# Events	Evts. Burst-1	Disk Space
Merger1	<input type="text" value="612"/>	<input type="text" value="205337"/>	<input type="text" value="205337"/>	<input type="text" value="32%"/>
Merger2	<input type="text" value="610"/>	<input type="text" value="152592"/>	<input type="text" value="152592"/>	<input type="text" value="18%"/>
Merger3	<input type="text" value="611"/>	<input type="text" value="86222"/>	<input type="text" value="86222"/>	<input type="text" value="35%"/>

Clock

10/27/2016 05:33:42 PM

PCFarm

Per burst sum

Detector	MEPs/Producer	Lost	Choke/Errors
LOTP	191515	0	0
KTAG	191235	6216	0
GTK	205336	1355	0
CHANTI	191515	0	0
LAV	191515	0	0
STRAW	191426	12888	0
RICH	191515	0	0
CHOD	191515	0	0
IRC SAC	191515	0	0
MUV3	191515	0	0
HAC	191515	0	0
MUV2	206691	0	0
MUV1	206691	0	0
LKR	206691	0	0

Process logging

- Logging utilities exist in artdaq
 - TRACE debugging tool, with message type markers and accurate time-stamps per “printf”-type statement
 - Message facility, with configurable message filtering/routing and message severity levels
- Plan to use “Elastic Stack” products to store, search, and visualize log messages
 - In use at CERN, with hooks to JCOP
 - Allow for customizable queries/searches across log files
 - Customizable visualization, with different types of data representations and support for time-series metrics

Configuration management

- *artdaq* configurations consist of three main parts
 - Architecture configuration
 - How many nodes, where, doing what?
 - Core software configuration
 - Core configuration for the BoardReaders, EventBuilders, Aggregators, and online monitoring process
 - Not dependent on experiment-specific hardware interfaces
 - “hardware” configuration
 - Configuration of the fragment generator processes that interface to hardware
- Current uses
 - Architecture configuration used at process launch time (mpirun)
 - Software and hardware configurations combined in FHiCL configuration source (one per process)

Configuration Management Tool

- *artdaq* team developing a common configuration management tool
 - MongoDB and file-system-based options for storing FHiCL configurations
 - Schema-less → very flexible/easy to change
 - Simple interfaces for uploading, downloading, and changing interfaces
 - Including graphical interface for changing single values
 - History stored in the database, and full configuration FHiCL available in final data files
 - *art* utilities allow storing of all FHiCL configurations throughout file history, so piggy-backing on that
- Will need to work out detailed interface to JCOP
 - Simplest method through same mechanisms as the run control, and maintain standalone configuration management tool

artdaq Configuration Editor (beta)

ARTDAQ Configuration Editor

Information ConnectConfig RunConfig HardwareConfig

dcm-1-01-01

File Information

File Version

Version001

Collection Name

HardwareConfig

Entity Name

dcm-1-01-01

Assigned Configurations

Name	Date Assigned
notprovided	Thu Oct 13 2016 10:30:38 GMT-0500 (CDT)
NovaConfig_1476372659725	Thu Oct 13 2016 10:30:59 GMT-0500 (CDT)
NovaConfig2	Fri Oct 14 2016 13:27:39 GMT-0500 (CDT)

File Contents

Name	Value	User Comment
▼ DcmConfiguration		
▼ HardwareConfiguration		
▼ FEBPixelEnableSet		
▼ FEBPixelEnable		
PixelMask	0xFFFFFFFF	
▼ FEBMask		
FEBMaskLo	0xFFFFFFFF	
FEBMaskHi	0x00000000	
DcmFPGAMode	FEBDCSMODE	

Component testing

- Run control/operational monitoring
 - Initial DIM server to DAQInterface and connection to JCOP system for process status and management in November 2016
 - Metric plugin to *artdaq* monitoring metrics in December 2016
- Configuration management
 - Initial testing versions exist and undergoing final refinement
 - Components expected to all be ready January 2017
- Testing plan:
 - Develop integrated system on *artdaq* demo project, and then translate tools to protodune DAQ as they are in final form
 - Expect *artdaq* demo demonstrator with JCOP and configuration management finalized in February 2017

Risks

- Immediate/urgent needs during commissioning to “get something working” lead to incoherent operation plan
 - Always a risk for downstream software components
 - Mitigating by...
 - Relying on solid, developed, scalable, flexible frameworks
 - Establishing system control/monitoring design with clear interfaces
 - Delivering working test system as template/guide for further development
 - Maintaining good communication across DAQ groups on needs and integration

Conclusion

- Have designs in place for run control, operational monitoring, and configuration
 - Exploiting many existing elements of *artdaq* system
 - Where we need something new or different...
 - Leveraging experience from 35-ton and other experiments to inform design
 - Leveraging CERN tools, resources, and support
- Designs will meet requirements/needs of ProtoDUNE for full-featured run control, monitoring, and configuration tools

Backup