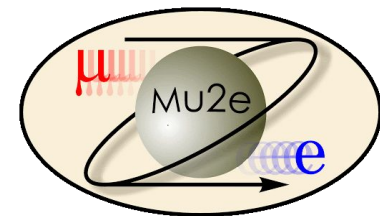# *DCS on otsdaq*

Antonio Gioiosa

Mu2e Italy General Meeting

September 03, 2020

otsdaq and Slow Control integration

Summary:

- *otsdaq* overview
- EPICS connection and plugin development in *otsdaq*
- Slow Controls Monitoring in *otsdaq*
  - Calorimeter monitoring and the Slow Controls GUI
- Integration with State Machine
  - Integration of *otsdaq* front-end DAQ hardware Process Variables (PVs) with EPICS (*DTC/ROCs/CFO*)
  - Integration of online data processing *output metrics* with EPICS (*artdaq deadtime/trigger rates/history*)
  - Integration of *otsdaq* DQM output metrics with EPICS
- Milestones

🔷 **Fermilab**

otsdaq overview

otsdaq is a Ready-to-Use data-acquisition (DAQ) solution aimed at test-beam, detector development, and other rapid-deployment scenarios. *otsdaq* uses the *artdaq* DAQ framework under-the-hood, providing flexibility and scalability to meet evolving DAQ needs. *otsdaq* provides a library of supported front-end boards and firmware modules which implement a custom UDP protocol. Additionally, an integrated Run Control GUI and readout software are provided, preconfigured to communicate with *otsdaq* firmware.

Last modified: 04/29/20 | email Fermilab
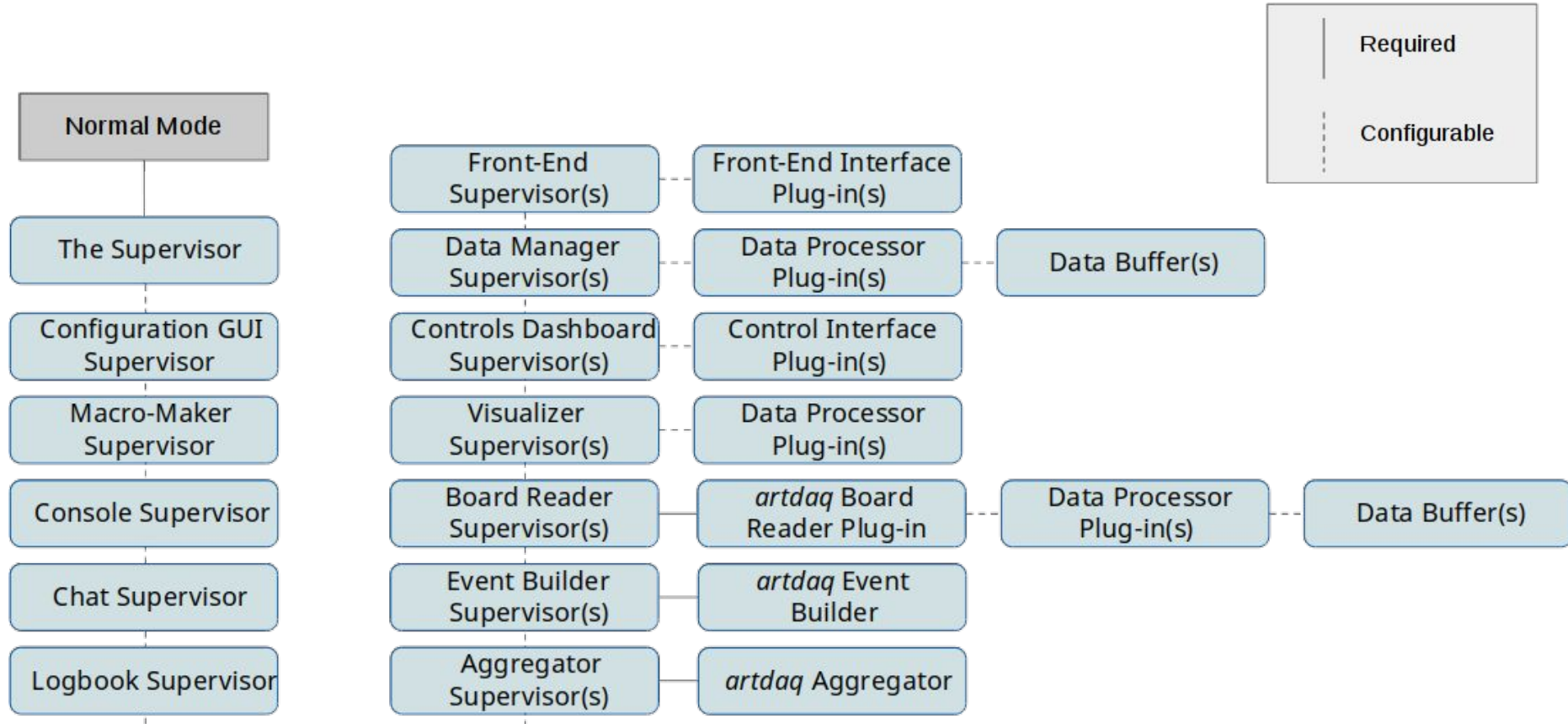
otsdaq overview

- *otsdaq* is a Ready-to-Use data-acquisition (DAQ) solution aimed at test-beam, detector development, and other rapid-deployment scenarios

- *otsdaq* uses the *artdaq* DAQ framework under-the-hood, providing flexibility and scalability to meet evolving DAQ needs

- *otsdaq* provides a library of supported front-end boards and firmware modules which implement a custom UDP protocol

- an integrated Run Control GUI and readout software are provided, preconfigured to communicate with *otsdaq* firmware
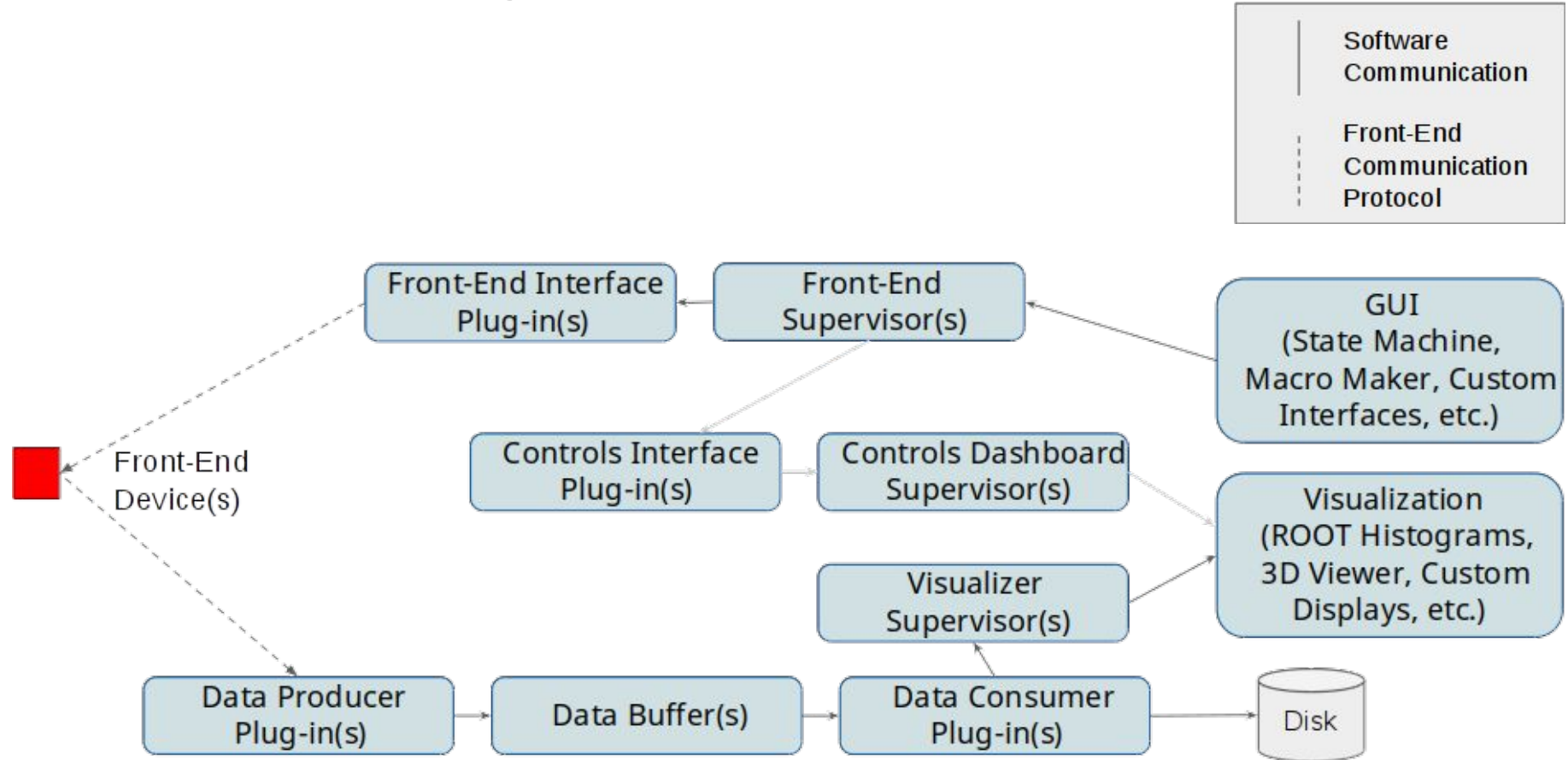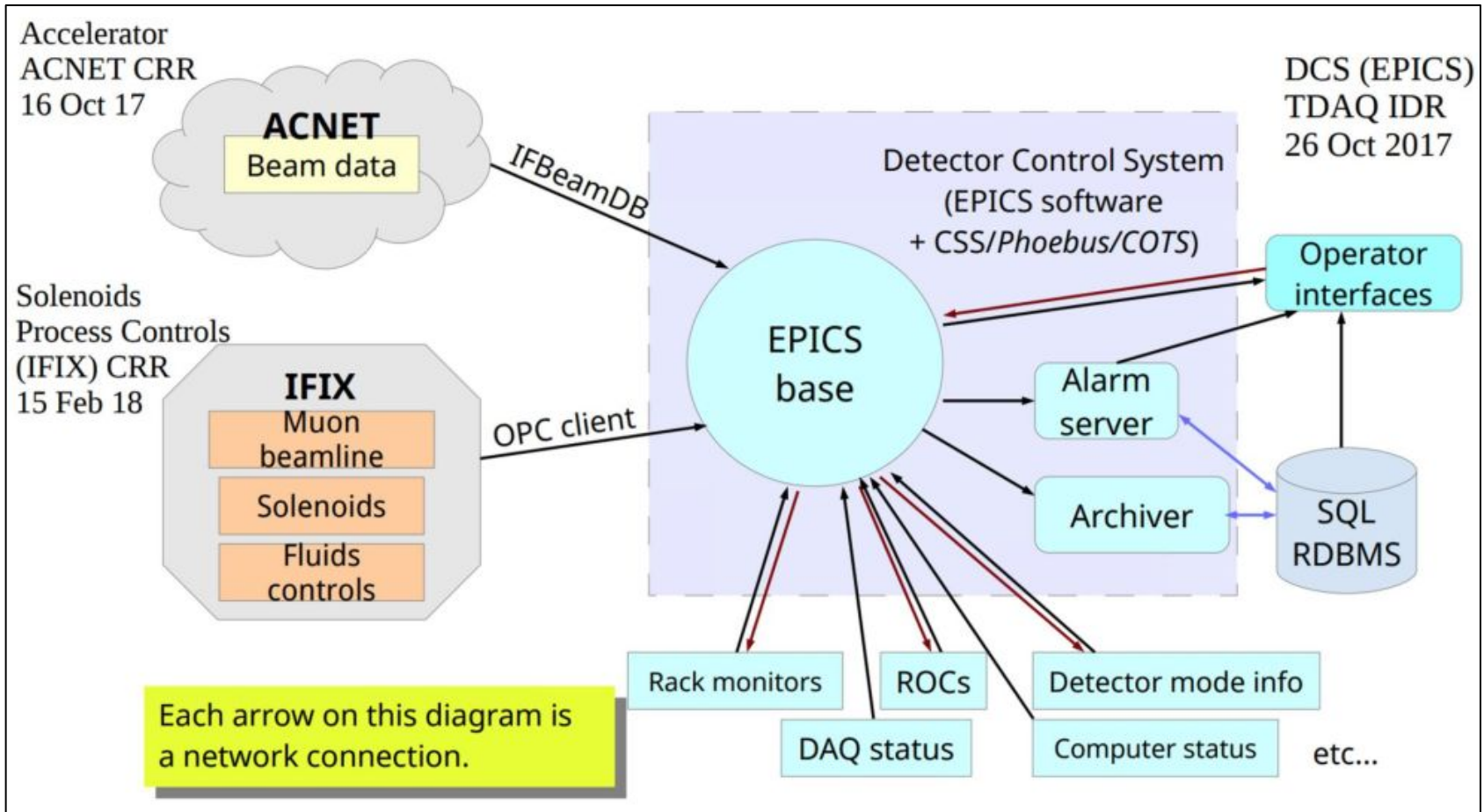
🔷 Fermilab

otsdaq overview

Components

- Normal Mode
  - The Supervisor
  - Configuration GUI Supervisor
  - Macro-Maker Supervisor
  - Console Supervisor
  - Chat Supervisor
  - Logbook Supervisor

- Front-End Supervisor(s) — Front-End Interface Plug-in(s)
- Data Manager Supervisor(s) — Data Processor Plug-in(s) — Data Buffer(s)
- Controls Dashboard Supervisor(s) — Control Interface Plug-in(s)
- Visualizer Supervisor(s) — Data Processor Plug-in(s)
- Board Reader Supervisor(s) — *artdaq* Board Reader Plug-in — Data Processor Plug-in(s) — Data Buffer(s)
- Event Builder Supervisor(s) — *artdaq* Event Builder
- Aggregator Supervisor(s) — *artdaq* Aggregator

Required (solid line)
Configurable (dashed line)

https://mu2e-docdb.fnal.gov/cgi-bin/private/ShowDocument?docid=14086

🔷 Fermilab

otsdaq overview

Data Flow Block Diagram

https://mu2e-docdb.fnal.gov/cgi-bin/private/ShowDocument?docid=14086

EPICS *connection and plugin development in* otsdaq

(See the slow controls web-app Mu2e Document 26941-v1 for details)

EPICS connection and plugin development in otsdaq

Channel subscription to EPICS

- Value

- Alarm (Status, Severity)

- Settings
  - *PV Unit, Lower and Upper Warning Limits, Lower and Upper Alarm Limit, Lower and Upper Control Limits, Lower and Upper Display Limits*

- Channel history and alarms retrieving from EPICS Archiver Databases
  - *dcs_archiver*
  - *dcs_alarm*
  - *dcs_log*
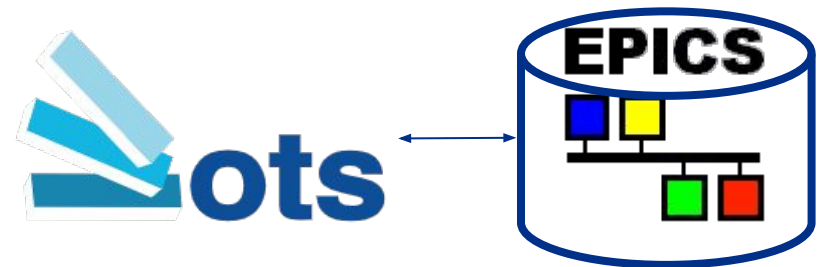
Slow Controls Monitoring in otsdaq

Slow Controls Software purpose

- Allow the user to monitor or interact with their own DAQ hardware. Able to see things such as:

  Alarms, Warnings, Readouts, Timestamps, Status

- Interact through a web interface that is:

  Lightweight, User-Friendly, Plug n' Play, Customizable

(Control System overview Mu2e Document 9082-v1)

🍀 **Fermilab**
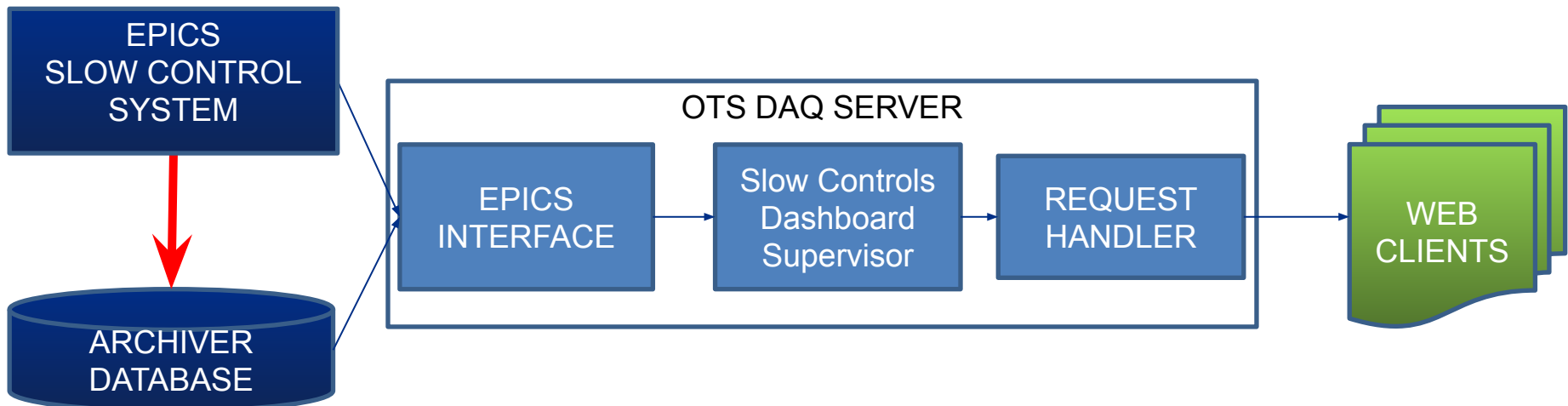
Slow Controls Monitoring in otsdaq

Slow Controls C++ Hierarchy

Controls Dashboard Supervisor(s) → xdaq Supervisor GUI server

Controls Interface Plug-in(s) →

ots::ControlsVInterface virtual Controls Interface

Configurable through *otsdaq*

::subscribe(pv name)
::unsubscribe(pv name)
::getList() //of pvs
::getCurrentValue(pv name)
::getSettings(pv name)

Epics Interface Plug-in →

ots::EpicsInterface Controls Interface
Implements virtual interface in EPICS style

https://mu2e-docdb.fnal.gov/cgi-bin/private/ShowDocument?docid=14086

🐝 Fermilab

Slow Controls Monitoring in otsdaq

Slow Controls GUI Hierarchy

https://mu2e-docdb.fnal.gov/cgi-bin/private/ShowDocument?docid=14086
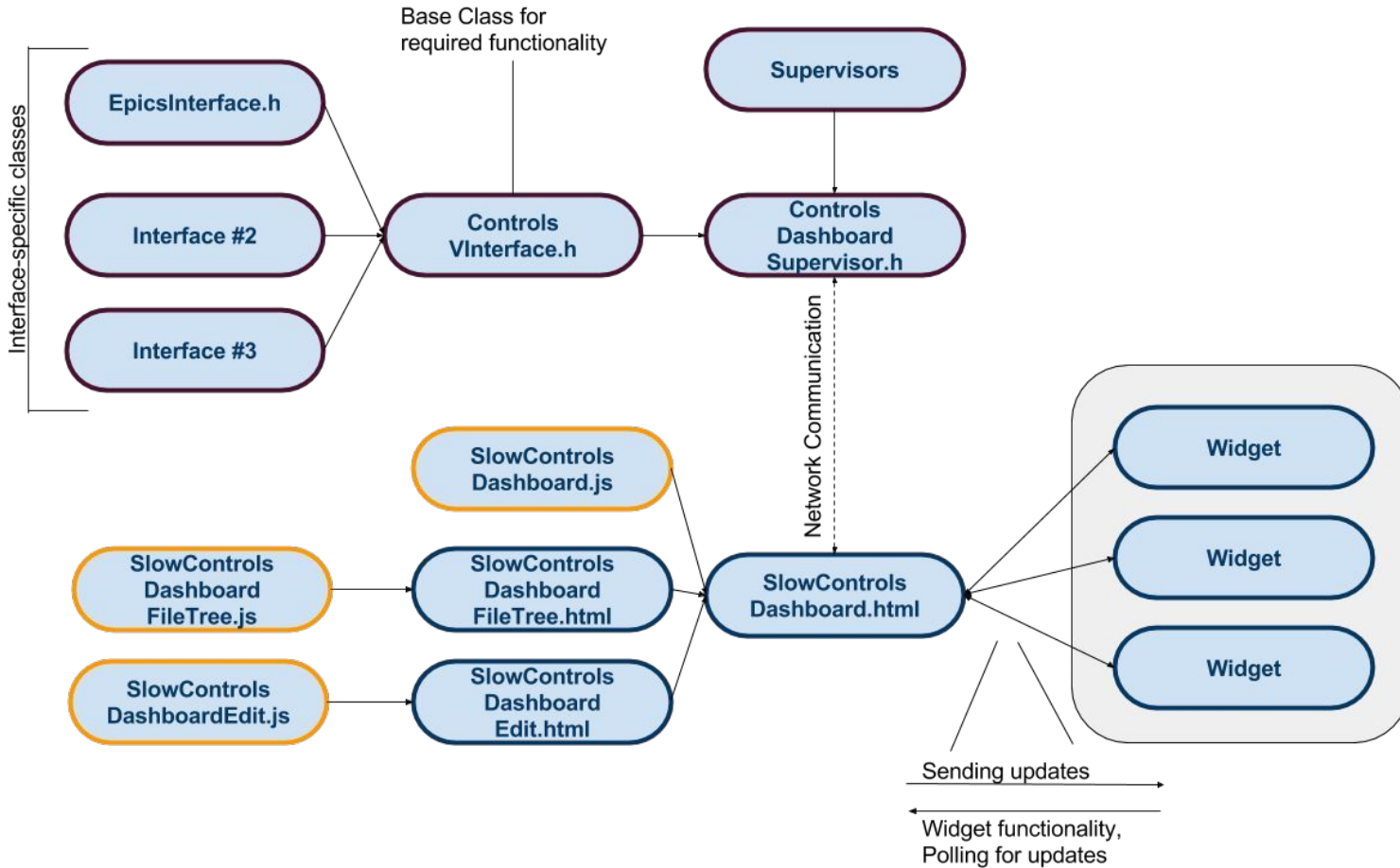
Slow Controls Monitoring in otsdaq

Configuring by specific tables in otsdaq

DesktopIconTable, XDAQApplicationPropertyTable, XDAQApplicationTable, XDAQContextTable

Configuration GUI in otsdaq desktop environment
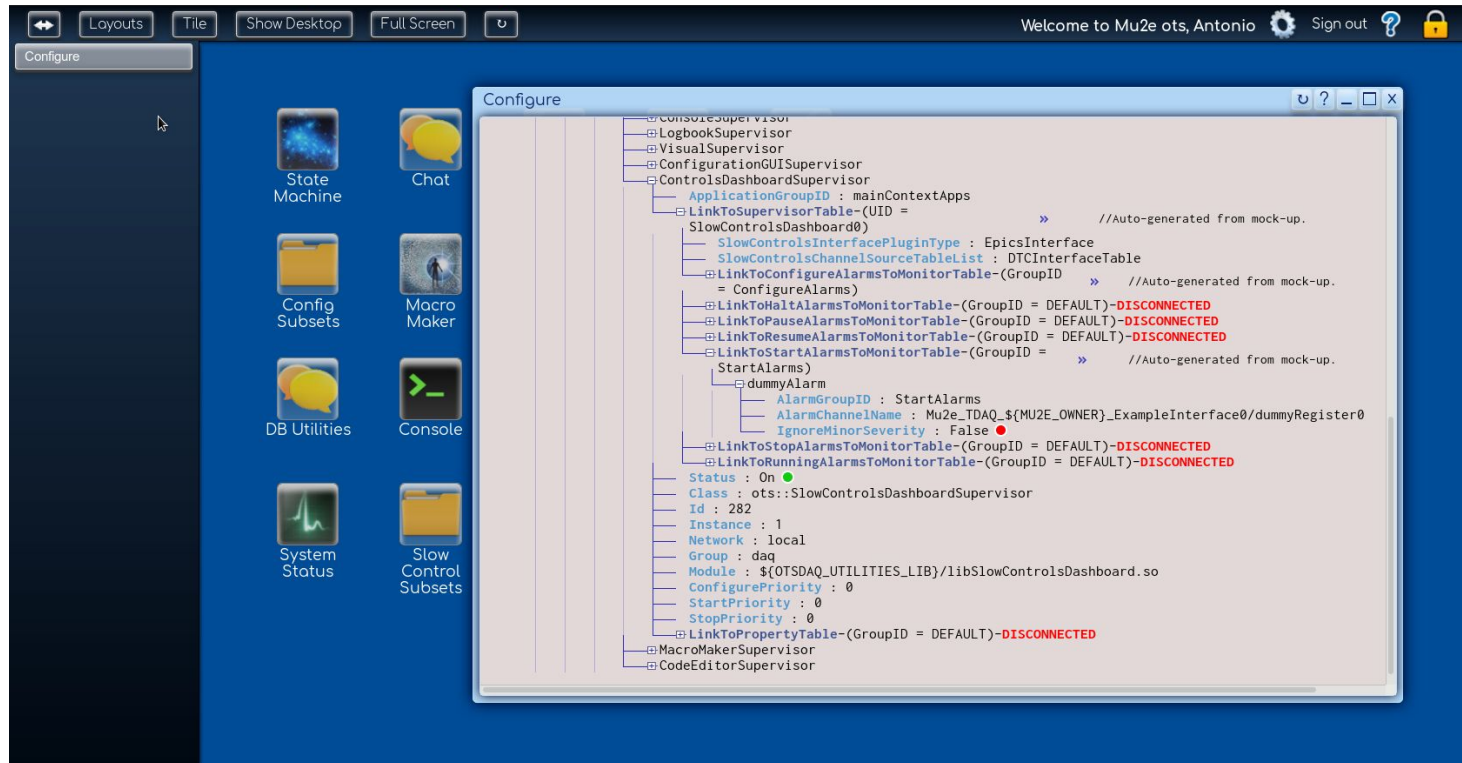
Instructions on EPICS configuration in otsdaq slow controls dashboard are in the redmine wiki at:

https://cdcvs.fnal.gov/redmine/projects/mu2e-dcs/wiki/EPICS_otsdaq_controls_dashboard

🟦 **Fermilab**

Slow Controls Monitoring in otsdaq

Basic Widget Mechanics

- All widgets have six required methods:

    init(), getParameters(), setParameters(), setupPVs(), newWidget(), and newValue()

    (see also Control System overview Mu2e Document 9082-v1)

Widget properties

- Dynamic sizing
- Proper handling of setups
- Value error, warning and alarm handling
- Disconnection handling

Load and save dashborad page in XML
Cs-Studio Phoebus compatible format
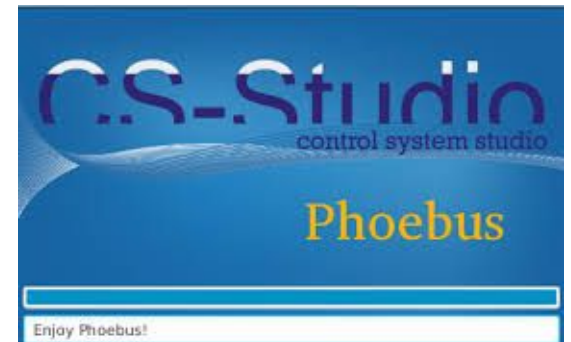
Slow Controls Monitoring in otsdaq

- Widgets implemented:

  text update, label, table, meter, strip chart, thermometer, siren alarm, 2D-stoplight, Root visualizer

- Extra libraries used:

  d3.js, Plot.ly, Tabulator.js

Example of page loading
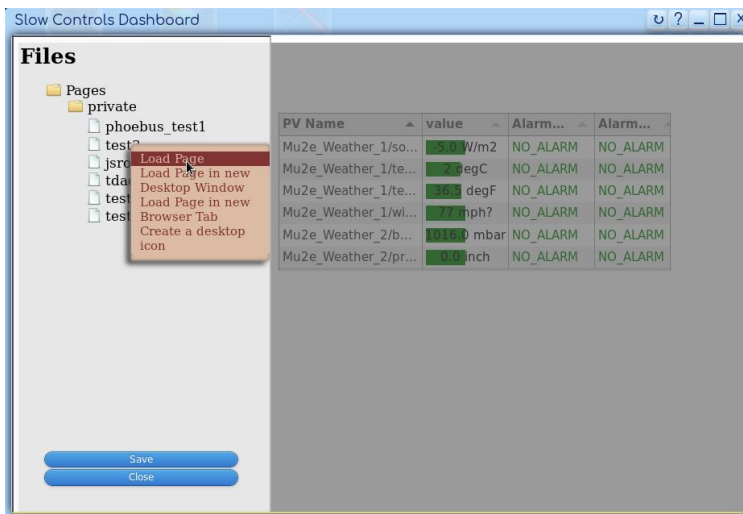
Example of loaded page
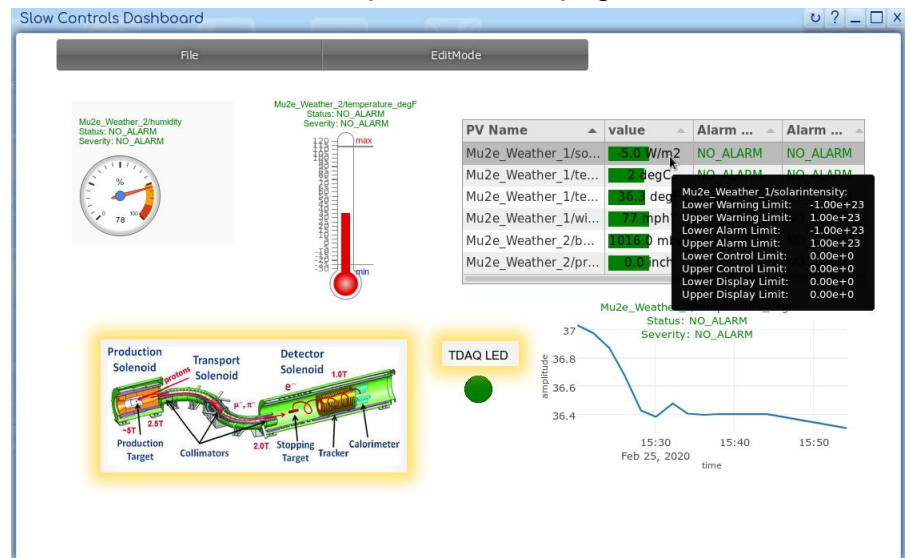
🔶 Fermilab

Slow Controls Monitoring in otsdaq

- ## Widgets implemented:

  text update, label, table, meter, strip chart, thermometer, siren alarm, 2D-stoplight, Root visualizer

- ## Extra libraries used:

  d3.js, Plot.ly, Tabulator.js

Editor
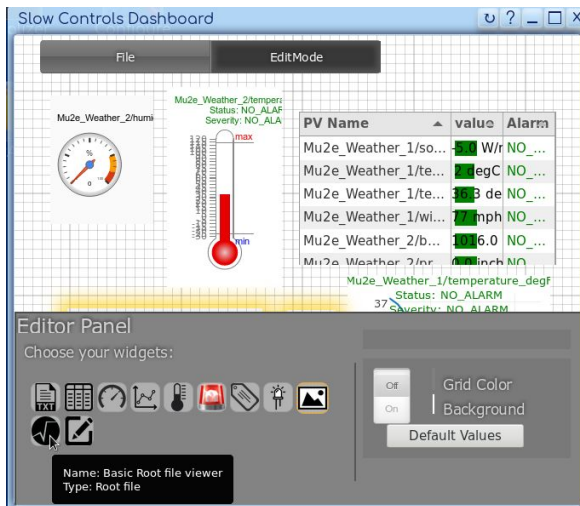
Example of widget settings window

Fermilab

Slow Controls Monitoring in *otsdaq*

new widget attributes editor

🔶 Fermilab
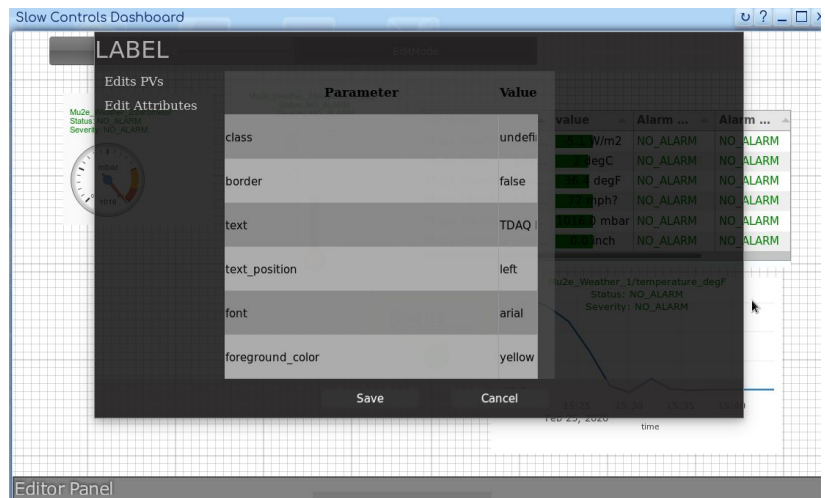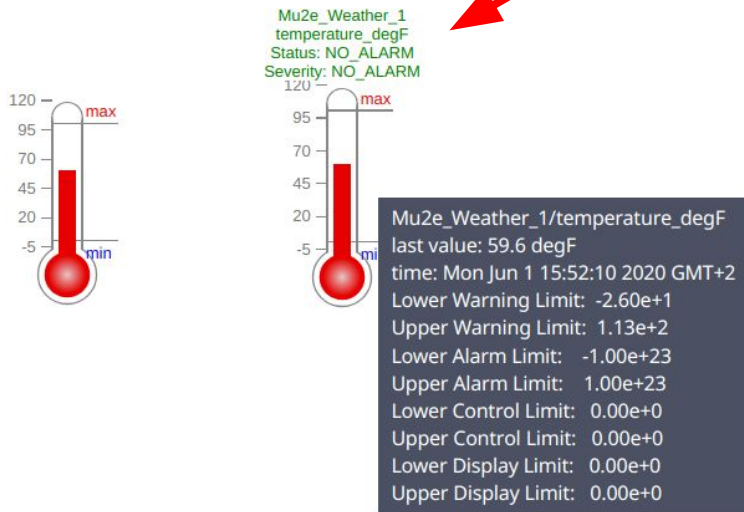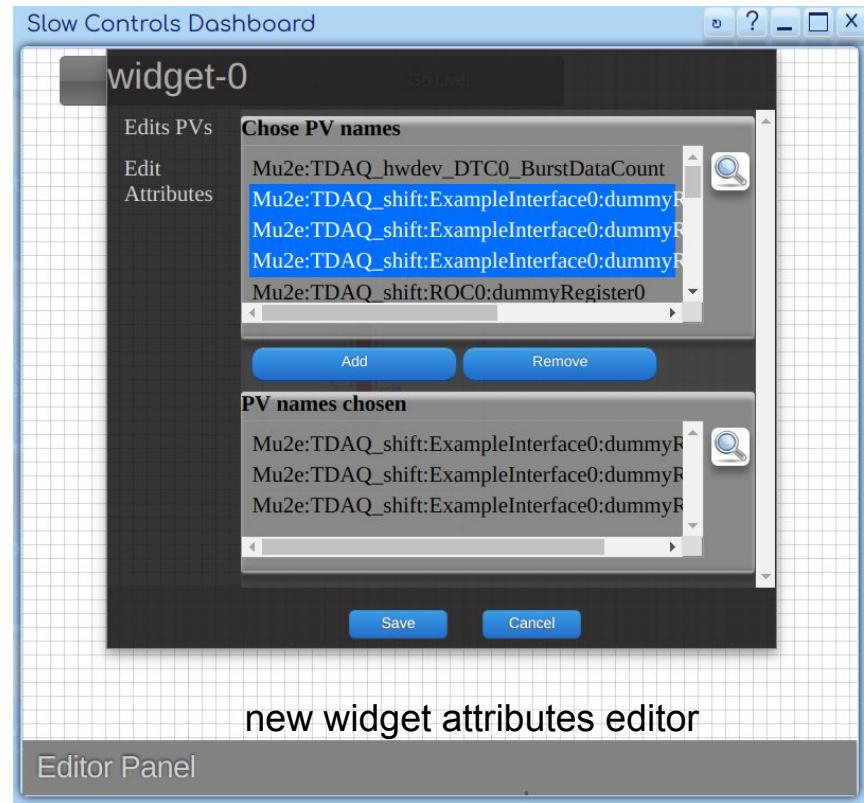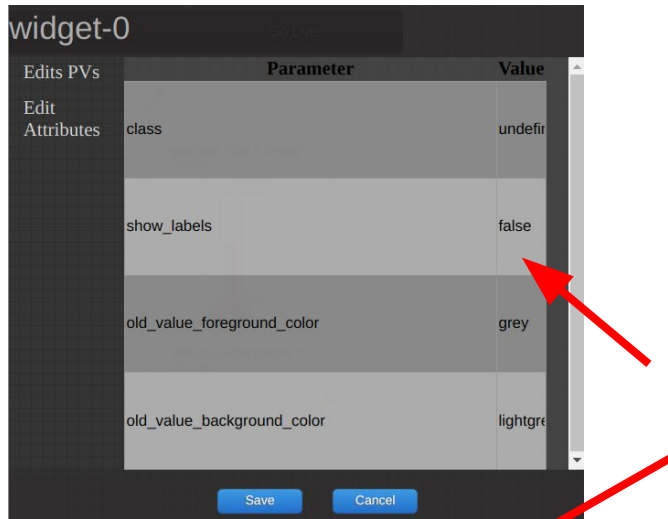
Slow Controls Monitoring in otsdaq
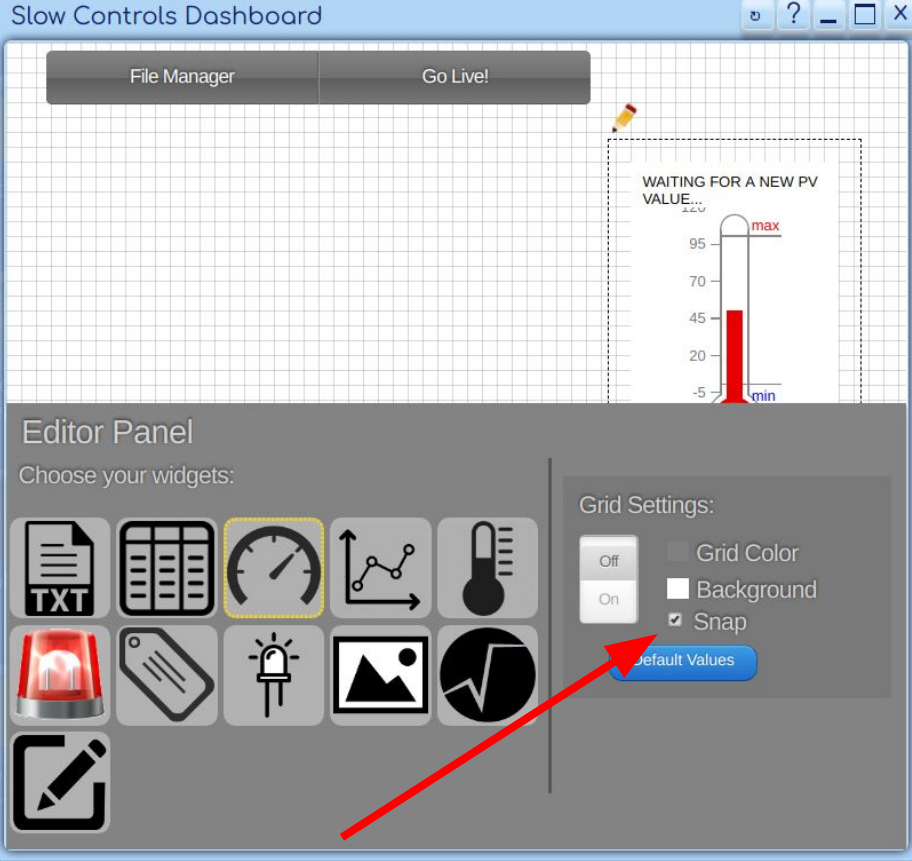
new Feature: snap to grid
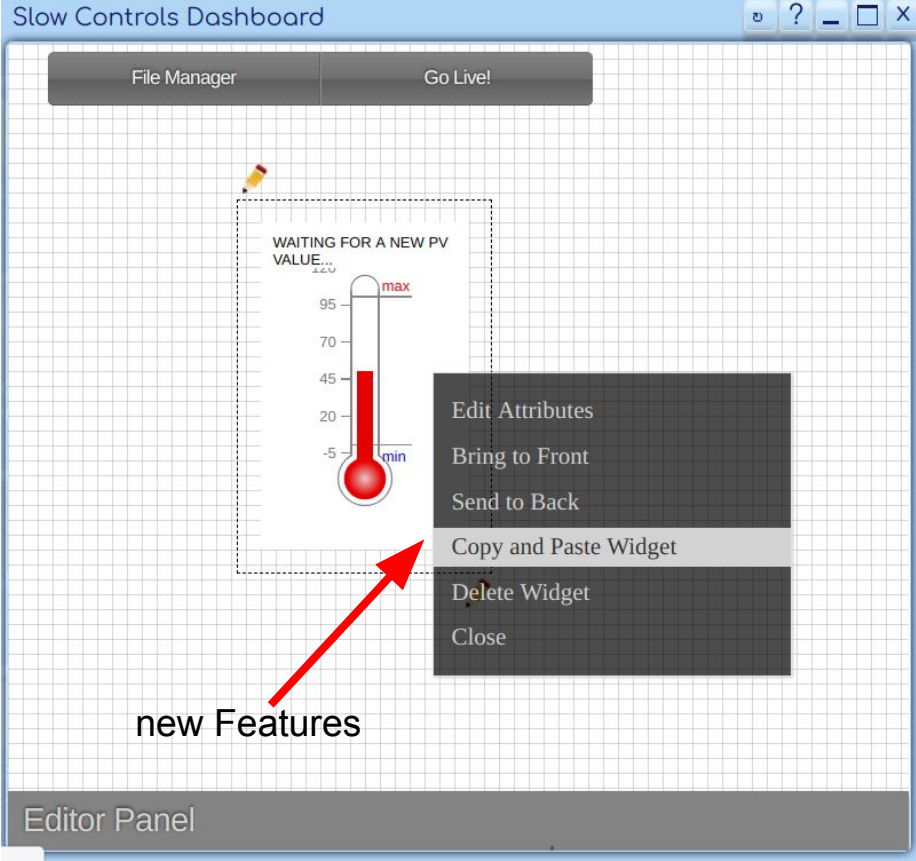
new Features
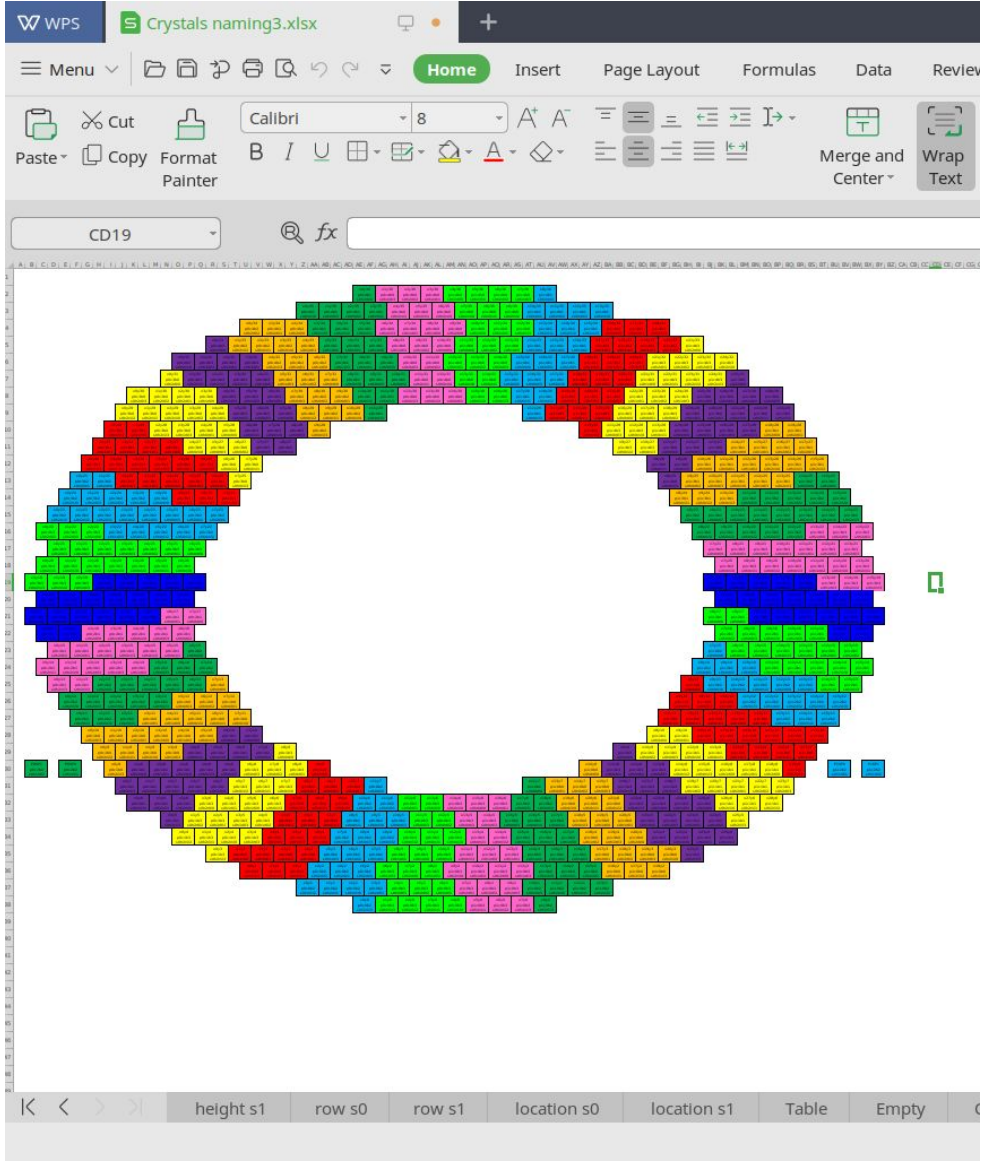
🎇 Fermilab

Calorimeter monitoring and the Slow Controls GUI

🐝 Fermilab

Calorimeter monitoring and the Slow Controls GUI

🔀 **Fermilab**
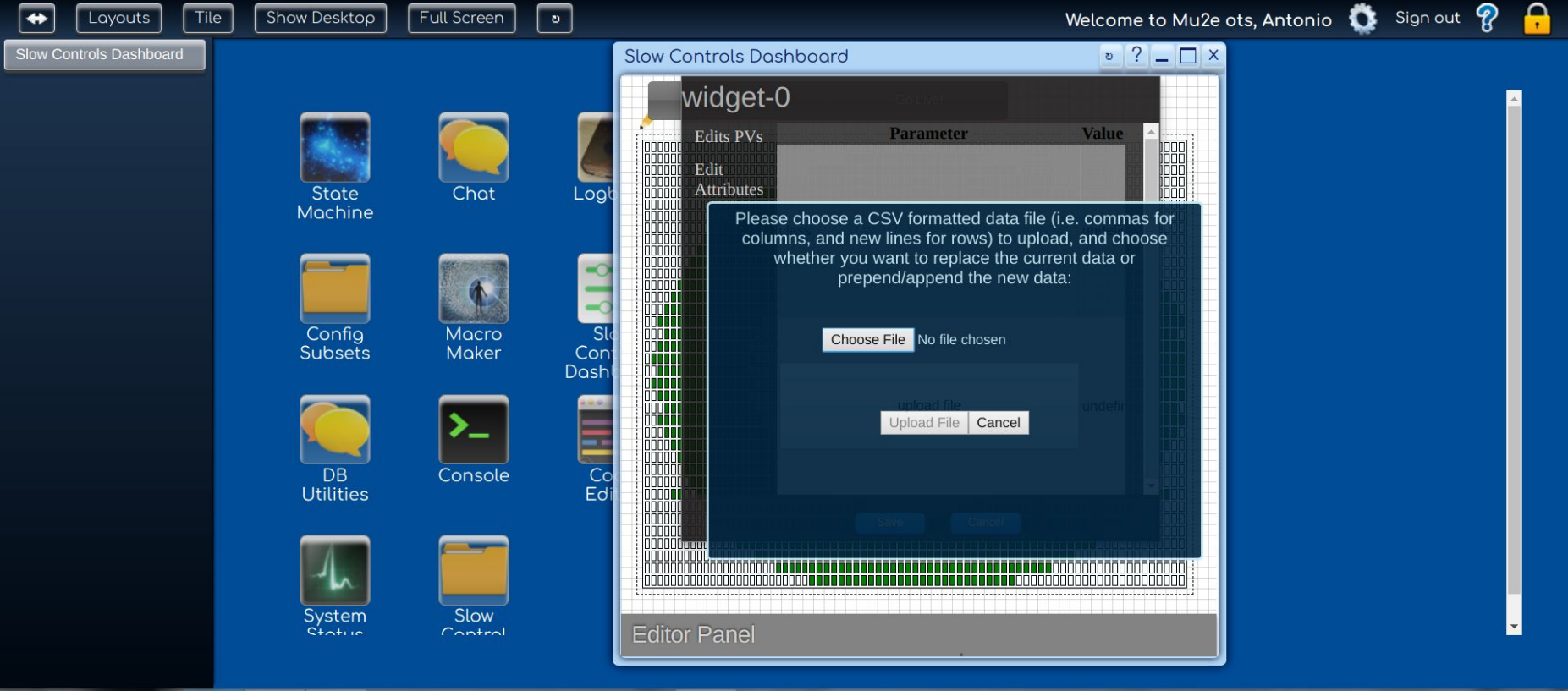
Calorimeter monitoring and the Slow Controls GUI
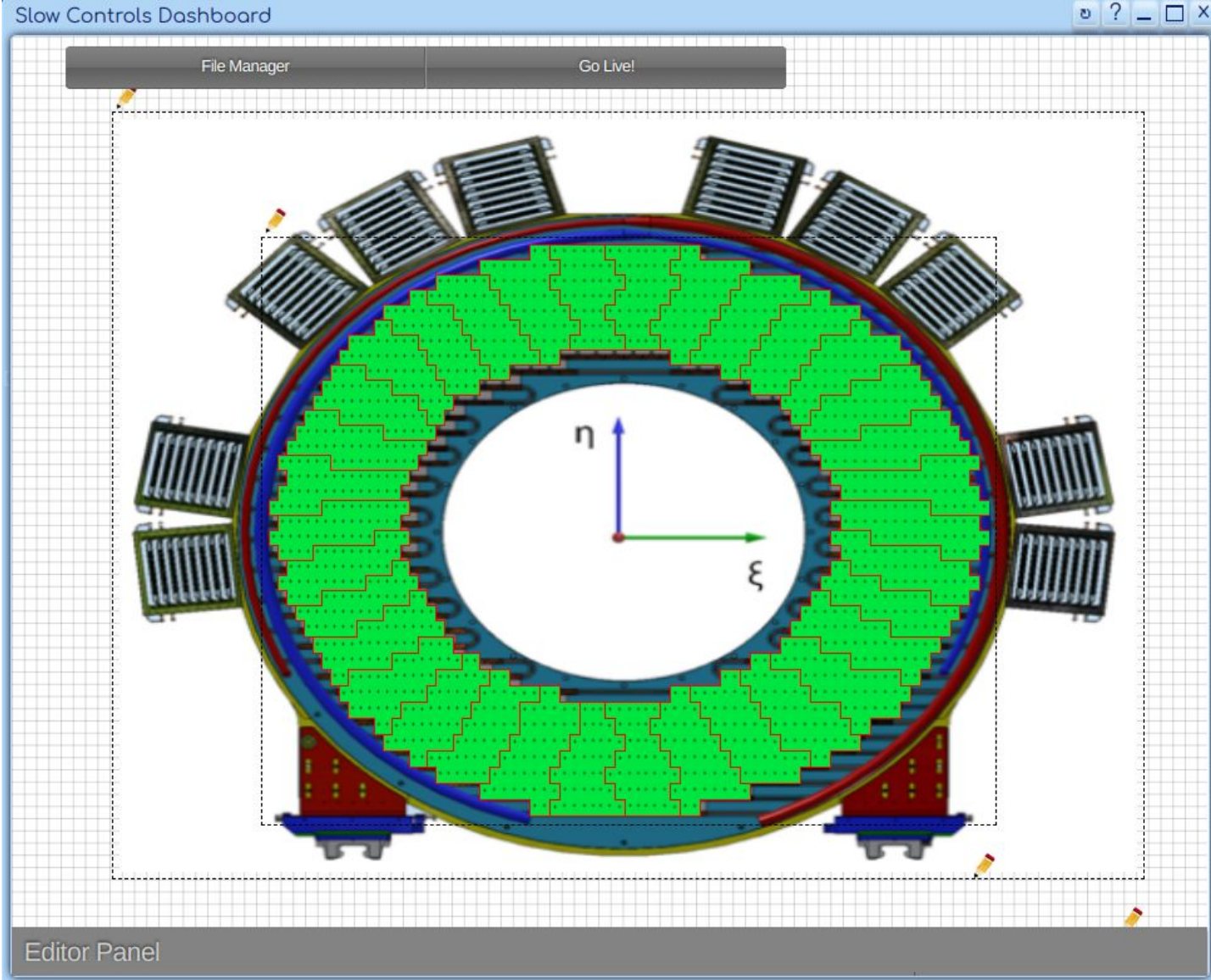
Slow Controls alarm notification by System Message

Configured by specific table in otsdaq:

AlarmAlertNotificationsTable
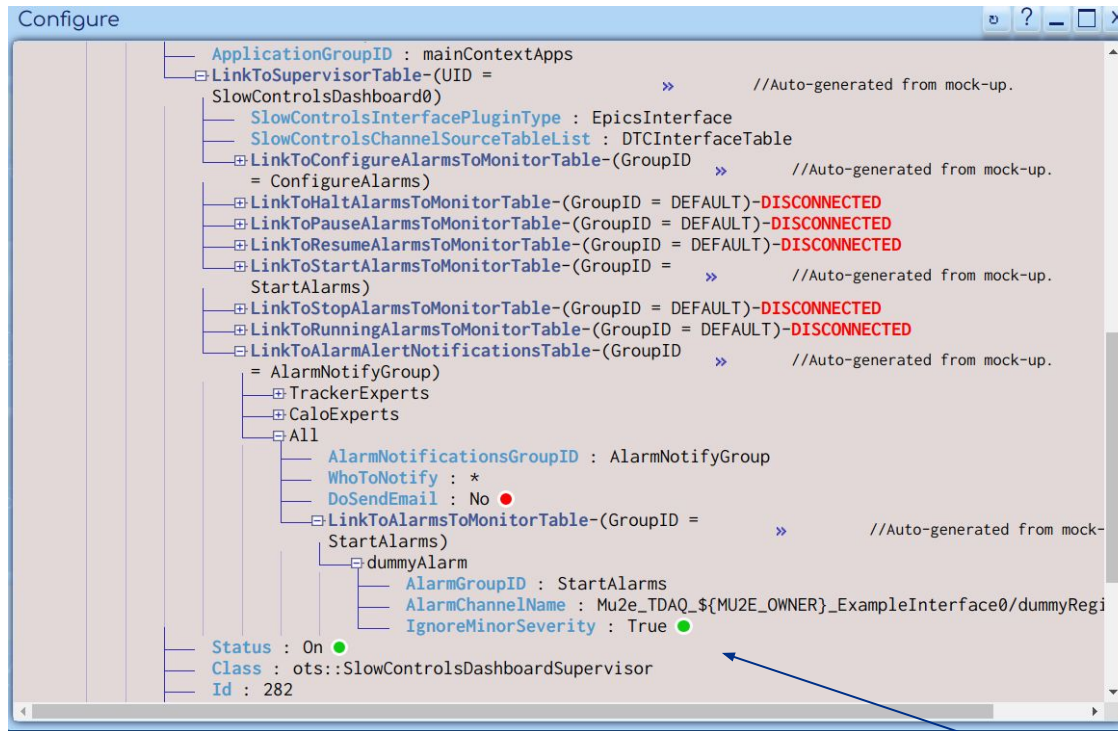
Ignore or not minor severity

Instructions are in:

https://mu2e-docdb.fnal.gov/cgi-bin/private/ShowDocument?docid=33299

Fermilab

Slow Controls alarm notification by System Message

System message alarm notification example

Instructions are in:

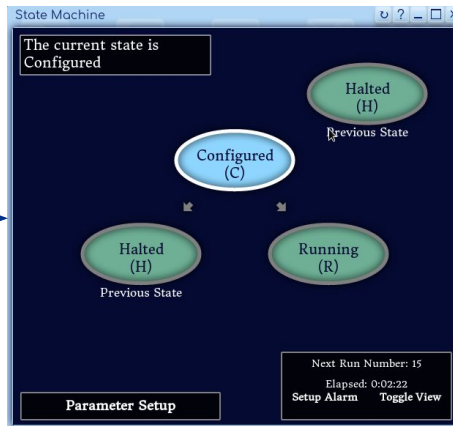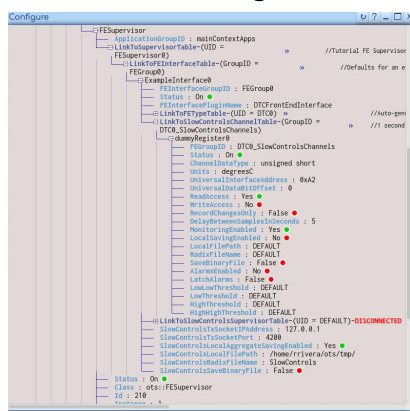https://mu2e-docdb.fnal.gov/cgi-bin/private/ShowDocument?docid=33299

🔷 **Fermilab**

Integration with State Machine

- State Machine Configuration and data subscription to EPICS
- Alarm propagation (from EPICS) and *otsdaq* State Machine handling

DAQ HW, artdaq and DQM metrics configuration

Alarm Configuration
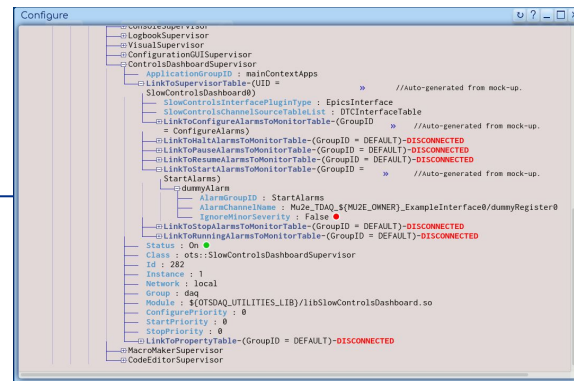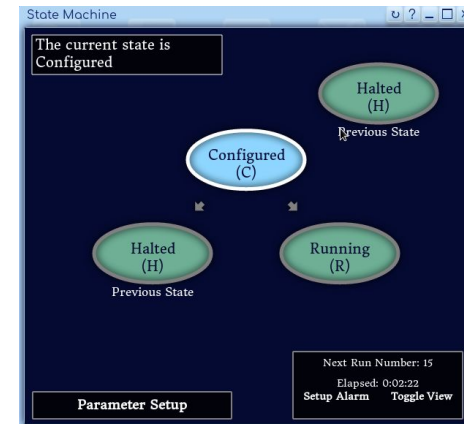

artdaq EPICS metrics Plugin

otsdaq EPICS Plugin

EPICS

🟦 Fermilab

Integration with State Machine

- Alarm propagation (from EPICS) and *otsdaq* state machine handling is available: needs just to identify which PV alarms, *status* and *severity* will be propagated

- *Tables and parameters designed for configuration*
  - SupervisorTable parameters:
    - *Slow Controls Interface Plugin Type*
    - *Slow Controls Channel Source Table List* (HW list i.e. DTC Interface, CFO Interface)

  - Alarms To Monitor Tables for transition to states:
    - *Configure*
    - *Halt*
    - *Pause*
    - *Resume*
    - *Start*
    - *Running*

Parameters

- *AlarmGroupID*
- *AlarmChannelName*
- *IgnoreMinorSeverity*

🔷 Fermilab

Integration with State Machine

- Alarm propagation (from EPICS) and *otsdaq* state machine handling: Example on "Start" transition

🔷 Fermilab

Integration of otsdaq front-end DAQ hardware with EPICS

Actions designed and developed in *otsdaq*

1. *otsdaq* DCS channels Front End and tables configuration

2. *otsdaq* State Machine configuration implementation

3. add/update channels info for IOC and Archiver DB

4. software IOC restarting

5. EPICS Archiver restarting

6. new *otsdaq* epics_plugin channels subscriptions to EPICS

7. Sending configured channels values to EPICS:

   otsdaq DCS channels new values → artdaq Metric Manager → software IOC → EPICS → otsdaq DCS GUI

🔷 **Fermilab**

Integration with State Machine

- *otsdaq* DTC channel slow controls settings → configuring SM → EPICS DBs and IOC → *otsdaq* CA subscription → Monitoring

🎛 Fermilab
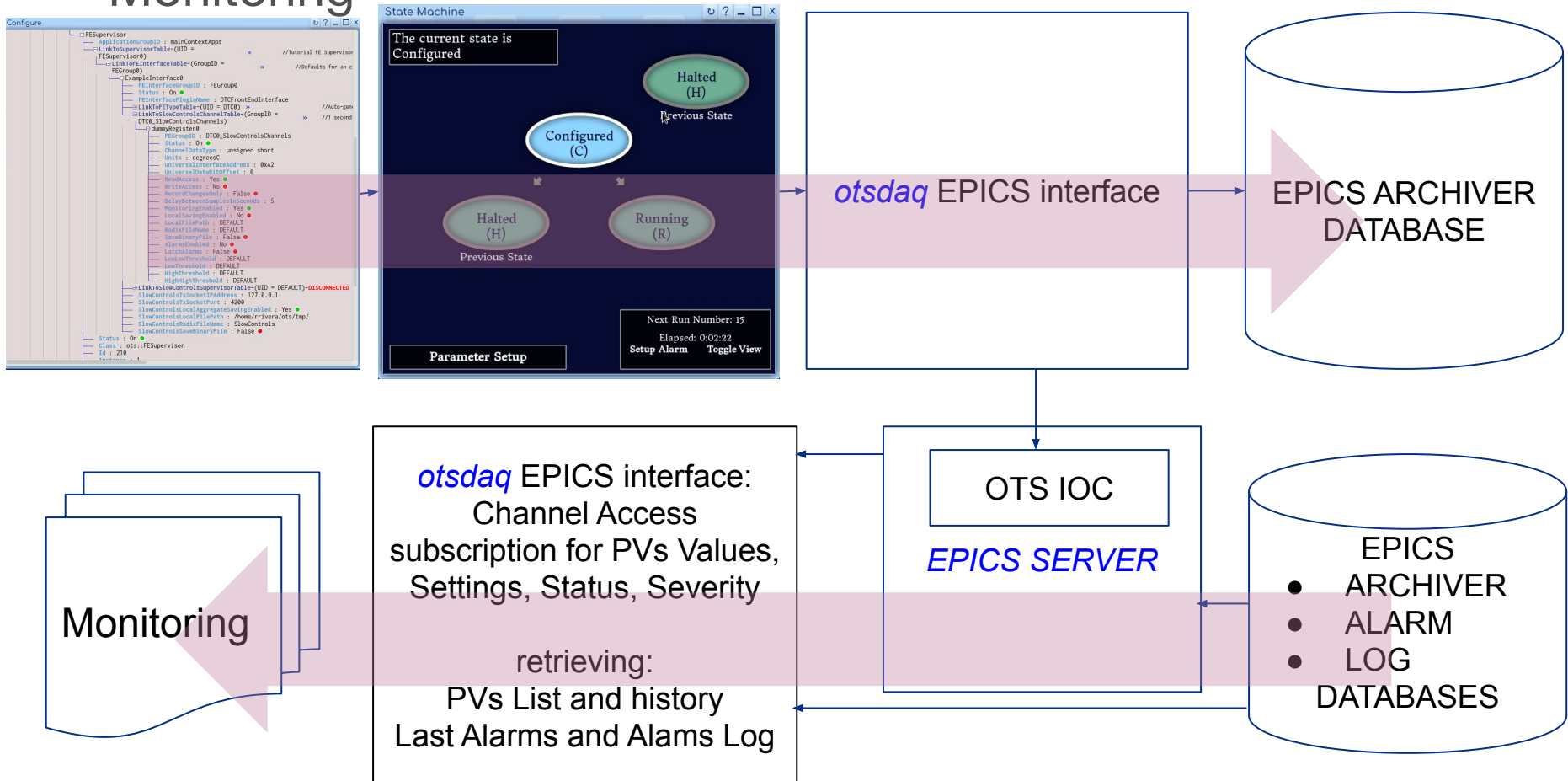
Online data processing output metrics and **EPICS**

scheduled

- *artdaq* tracks a large number of metrics about event rate and dataflow, which can be enabled at the metric plugin level
- the user can send a subset of metrics to EPICS
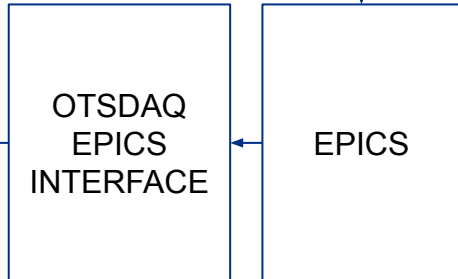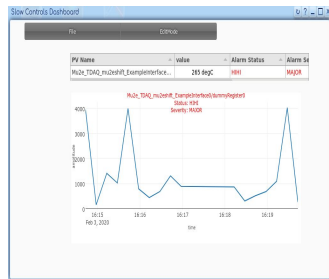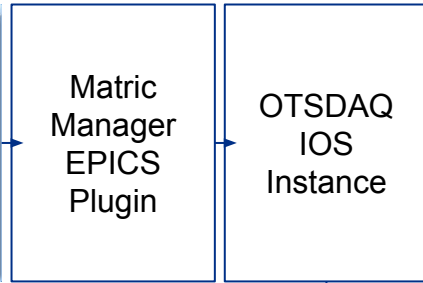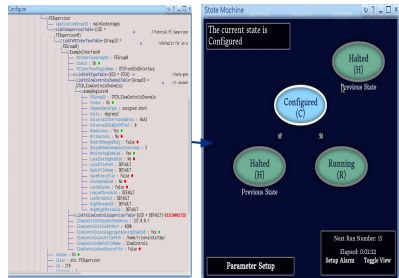
Artdaq Builtin Metric Inventory

Notes: %{RANK} will be replaced with appropriate ranks. Some plugins may perform transformations on these names, for example, graphite_metric replaces spaces with underscores.

artdaq_utilities

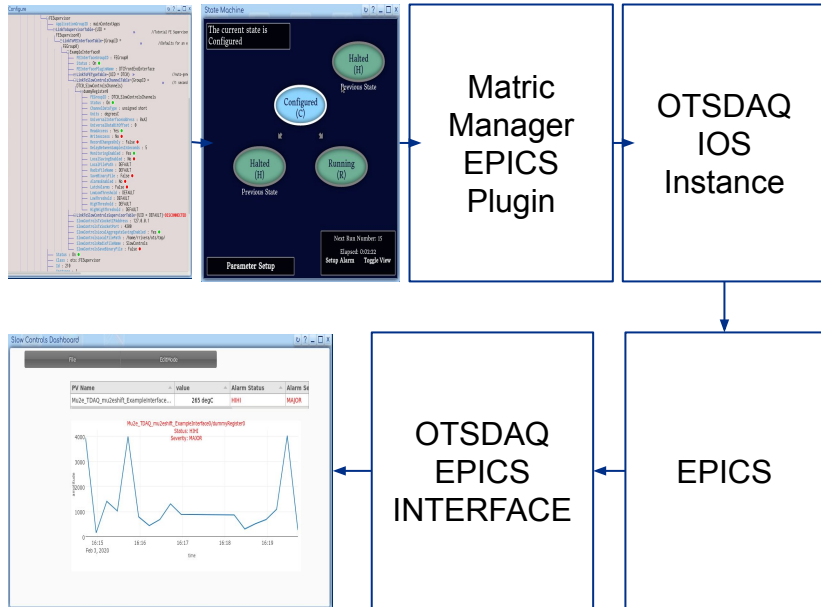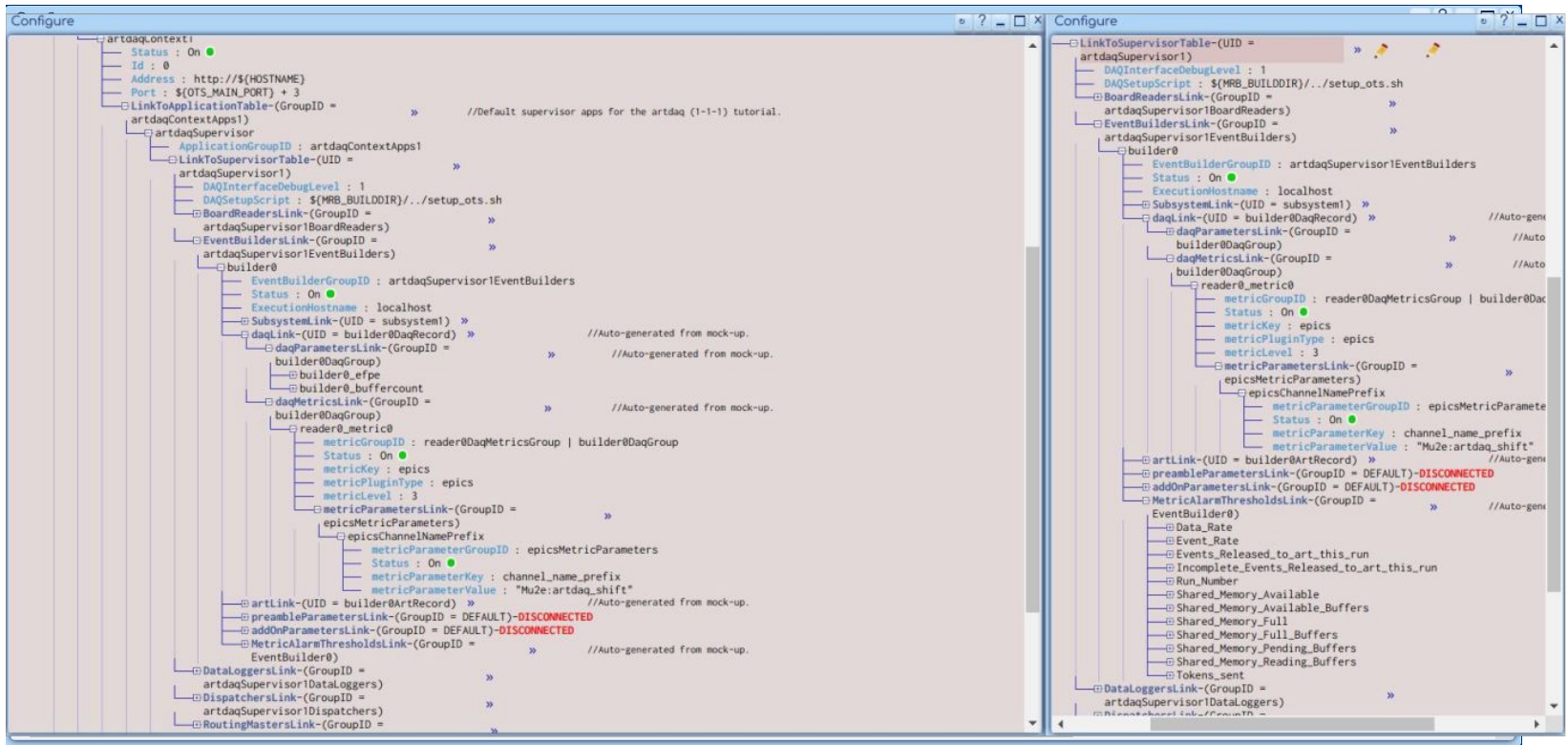| Name	Level	Units	Mode(s)	Source Class	Notes
Metric Calls	4	metrics	AccumulateAndRate	MetricManager	Tracks the number of sendMetric calls since the last time the metric dispatch thread ran
Missed Metric Calls	4	metrics	AccumulateAndRate	MetricManager	Number of discarded sendMetric calls due to queue overflow

artdaq

Name	Level	Units	Mode(s)	Source Class	Notes
Fragment Count	1	fragments	LastPoint	BoardReaderCore	
Fragment Rate	1	fragments/sec	Average	BoardReaderCore	
Average Fragment Size	2	bytes/fragment	Average	BoardReaderCore	
Data Rate	2	bytes/sec	Average	BoardReaderCore	
Avg Input Wait Time	3	seconds/fragment	Average	BoardReaderCore	Amount of time spent in CommandableFragmentGenerator::getNext
Avg BoardReader Sync Wait Time	3	seconds/fragment	Average	BoardReaderCore	Currently unused (always 0)
Avg Output Wait Time	3	seconds/fragment	Average	BoardReaderCore	Amount of time spent in DataSenderManager::sendFragment
Avg Frags Per Read	4	fragments/read	Average	BoardReaderCore	
DAQ Transition Time	4	s	Accumulate	Commandable	Records the amount of time spent in transitions
Average Sender Acknowledgement Time	3	seconds	Average	RoutingMasterCore	
Avg Table Acknowledge Time	3	seconds	Average	RoutingMasterCore	
Receiver Token Rate	1	updates/sec	Average	RoutingMasterCore	
Table Update Rate	1	updates/sec	Average	RoutingMasterCore	
Total Receiver Token Wait Time	3	seconds	Average	RoutingMasterCore	
Receiver Token Count	1	updates	LastPoint	RoutingMasterCore	
Table Update Count	1	updates	LastPoint	RoutingMasterCore	
bytesRead	3	B	LastPoint	SharedMemoryReader	
Avg Input Wait Time	3	s	Average	SharedMemoryReader	
Avg Processing Time	2	s	Average	SharedMemoryReader	
Avg Read Time	3	s	Average	SharedMemoryReader	
queue%Used	5	%	LastPoint	SharedMemoryReader	
Data Receive Size From Rank %{RANK}	5	B	Accumulate	DataReceiverManager	
Data Receive Time From Rank %{RANK}	5	s	Accumulate	DataReceiverManager	
Header Receive Size From Rank %{RANK}	5	B	Accumulate	DataReceiverManager	
Header Receive Time From Rank %{RANK}	5	s	Accumulate	DataReceiverManager	
Total Receive Size From Rank %{RANK}	5	B	Accumulate	DataReceiverManager	
Total Receive Time From Rank %{RANK}	5	s	Accumulate	DataReceiverManager	
Total Shared Memory Wait Time From Rank %{RANK}	3	s	Accumulate	DataReceiverManager	
Avg Fragment Wait Time From Rank %{RANK}	3	s	Average	DataReceiverManager	
Avg Shared Memory Wait Time From Rank %{RANK}	3	s	Average	DataReceiverManager	
Data Receive Rate From Rank %{RANK}	5	B/s	Average	DataReceiverManager	
Header Receive Rate From Rank %{RANK}	5	B/s	Average	DataReceiverManager	
Total Receive Rate From Rank %{RANK}	5	B/s	Average	DataReceiverManager	
Data Receive Count From Rank %{RANK}	3	fragments	LastPoint	DataReceiverManager	
Data Send Size to Rank %{RANK}	5	B	Accumulate	DataSenderManager	
Data Send Time to Rank %{RANK}	5	s	Accumulate	DataSenderManager	
Data Send Rate to Rank %{RANK}	5	B/s	Average	DataSenderManager	
Routing Wait Time	2	s	Average	DataSenderManager	
Data Send Count to Rank %{RANK}	3	fragments	LastPoint	DataSenderManager	
Routing Table Size	2	events	LastPoint	DataSenderManager	
Request Response Time	2	seconds	Average	RequestReceiver	
Average Event Size	1	Bytes	Average	SharedMemoryEventManager	
Tokens sent	2	Tokens	LastPoint	SharedMemoryEventManager	
Events Released to art this run	1	Events	LastPoint	SharedMemoryEventManager	
Incomplete Event Count	1	events	LastPoint	SharedMemoryEventManager	
Incomplete Events Released to art this run	1	Events	LastPoint	SharedMemoryEventManager	
Pending Event Count	1	events	LastPoint	SharedMemoryEventManager	
Run Number	1	Run	LastPoint	SharedMemoryEventManager	
Shared Memory Available %	2	%	LastPoint	SharedMemoryEventManager	
Shared Memory Available Buffers	2	buffers	LastPoint	SharedMemoryEventManager	
Shared Memory Full %	2	%	LastPoint	SharedMemoryEventManager	
Shared Memory Full Buffers	2	buffers	LastPoint	SharedMemoryEventManager	
Shared Memory Pending Buffers	2	buffers	LastPoint	SharedMemoryEventManager	
Shared Memory Reading Buffers	2	buffers	LastPoint	SharedMemoryEventManager	
Incomplete Event Rate	3	events/s	Rate	SharedMemoryEventManager	
Event Rate	1	Events/s	Rate	SharedMemoryEventManager	
Input Fragment Rate	1	Fragments/s	Rate	SharedMemoryEventManager	
Avg Data Acquisition Time	3	s	Average	CommandableFragmentGenerator	
Buffer Depth Bytes	1	bytes	LastPoint	CommandableFragmentGenerator	
Buffer Depth Fragments	1	fragments	LastPoint	CommandableFragmentGenerator	
Last Timestamp	1	Ticks	LastPoint	CommandableFragmentGenerator	

Matric Manager EPICS Plugin → OTSDAQ IOS Instance → EPICS → OTSDAQ EPICS INTERFACE → Slow Controls Dashboard

https://cdcvs.fnal.gov/redmine/projects/artdaq-utilities/wiki/Artdaq_Builtin_Metric_Inventory

🐸 **Fermilab**

Online data processing output metrics and EPICS

scheduled

- Classes to read info from *board reader, logger, dispatcher, routing master*, are completed and working. some metrics are in the configuration and tests are going on

https://cdcvs.fnal.gov/redmine/projects/artdaq-utilities/wiki/Artdaq_Builtin_Metric_Inventory

🛠 Fermilab

Online data processing output metrics and EPICS

Classes to read info from *board reader, logger, dispatcher, routing master*, are completed and working. some metrics are in the configuration and tests are going on

Example of artdaq metric tables configuration

https://cdcvs.fnal.gov/redmine/projects/artdaq-utilities/wiki/Artdaq_Builtin_Metric_Inventory

🎇 **Fermilab**

Online data processing output metrics and EPICS

Classes to read info from *board reader, logger, dispatcher, routing master*, are completed and working. some metrics are in the configuration and tests are going on

Example of artdaq metric tables configuration

https://cdcvs.fnal.gov/redmine/projects/artdaq-utilities/wiki/Artdaq_Builtin_Metric_Inventory

🎋 **Fermilab**

Online data processing output metrics and EPICS

Configuration and monitoring ready

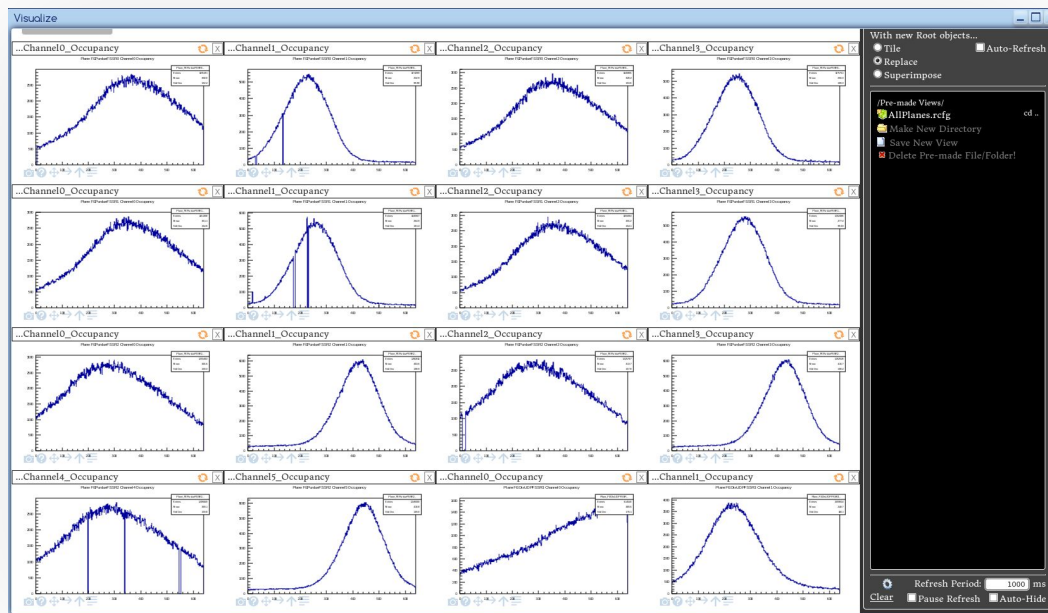| Slow Controls Dashboard			
PV Name	**value**	**Alarm Status**	**Alarm Sever**
Mu2e:artdaq_shift:builder0:Event_Rate	NF DC'd	N/a	N/a
Mu2e:artdaq_shift:builder0:Events_Released_to_art_this_run	NF DC'd	N/a	N/a
Mu2e:artdaq_shift:builder0:Run_Number	NF DC'd	N/a	N/a
Mu2e:artdaq_shift:builder0:Shared_Memory_Available	NF DC'd	N/a	N/a
Mu2e:artdaq_shift:builder0:Shared_Memory_Available_Buffers	NF DC'd	N/a	N/a
Mu2e:artdaq_shift:builder0:Shared_Memory_Full	NF DC'd	N/a	N/a
Mu2e:artdaq_shift:builder0:Shared_Memory_Full_Buffers	NF DC'd	N/a	N/a
Mu2e:artdaq_shift:builder0:Shared_Memory_Pending_Buffers	NF DC'd	N/a	N/a
Mu2e:artdaq_shift:builder0:Shared_Memory_Reading_Buffers	NF DC'd	N/a	N/a
Mu2e:artdaq_shift:builder0:Tokens_sent	NF DC'd	N/a	N/a

https://cdcvs.fnal.gov/redmine/projects/artdaq-utilities/wiki/Artdaq_Builtin_Metric_Inventory

🎟 **Fermilab**

Integration of otsdaq DQM output metrics with EPICS

Scheduled

- Data processing is the primary responsibility of the online DAQ. Mu2e's event window data will be processed through artdaq modules

- Data processor plugins is provided by otsdaq core

- DQM metrics are distinct from event rate and data flow metrics (which artdaq provides)

- DQM generates data products that are sent to an artdaq Dispatcher, which aggregates DQM metrics and presents them to a visualizer application

- Define which metrics send to EPICS

🟦 **Fermilab**

otsdaq and Epics Interface Documentation

- *Doc-db 33299*

  "Online DAQ and EPICS Interface" Documentation

- *redmine wiki*

  https://cdcvs.fnal.gov/redmine /projects/mu2e-dcs/wiki/EPIC S_otsdaq_controls_dashboard

FERMILAB
P.O. Box 500
Batavia, IL 60510-0500

the **Mu2e** project

Mu2e Project Document No.
33299

Fermilab./Group or Supplier/Contractor Document No.

EDMS Document No.

Date: 2020-May-07

Online DAQ and EPICS Interface

Abstract
Slow Controls, for the Mu2e experiment, provides the primary path for users to monitor their detector hardware. Slow Controls provides several mechanisms for monitoring, including the following: Alarms, Warnings, Readouts, Timestamps, Status. This document describes the design, how to further develop, and how to use the EPICS slow control interface in the online DAQ.

Prepared by:	Checked by:	Approved by:
A. Gioiosa R. Rivera E. Flumerfelt G. Horton-Smith		

Circulated to:

2020 - 2021 Milestones ?

Expected milestones

- Advanced Slow Controls Monitoring in *otsdaq*

- Real configuration to send *otsdaq* front-end DAQ hardware, data processing and DQM in EPICS

- Real configuration and Integration with State Machine

- Integration test on hardware

- connection with *otsdaq* @ mu2e building

🎇 Fermilab