DAQ Monitoring

Marco Roda

DUNE UK Meeting

4 July 2023
Bristol

Overview

- Monitoring goals and design
- Status and plans

- All deliverables presented here are essentially responsibility of the UK group
  - Mostly me
  - Contributions from other WGs as monitoring needs to be tailored to the specifics of their activity
    - Mostly CoreSW, SW Coordination and Infrastructure

What are we monitoring?

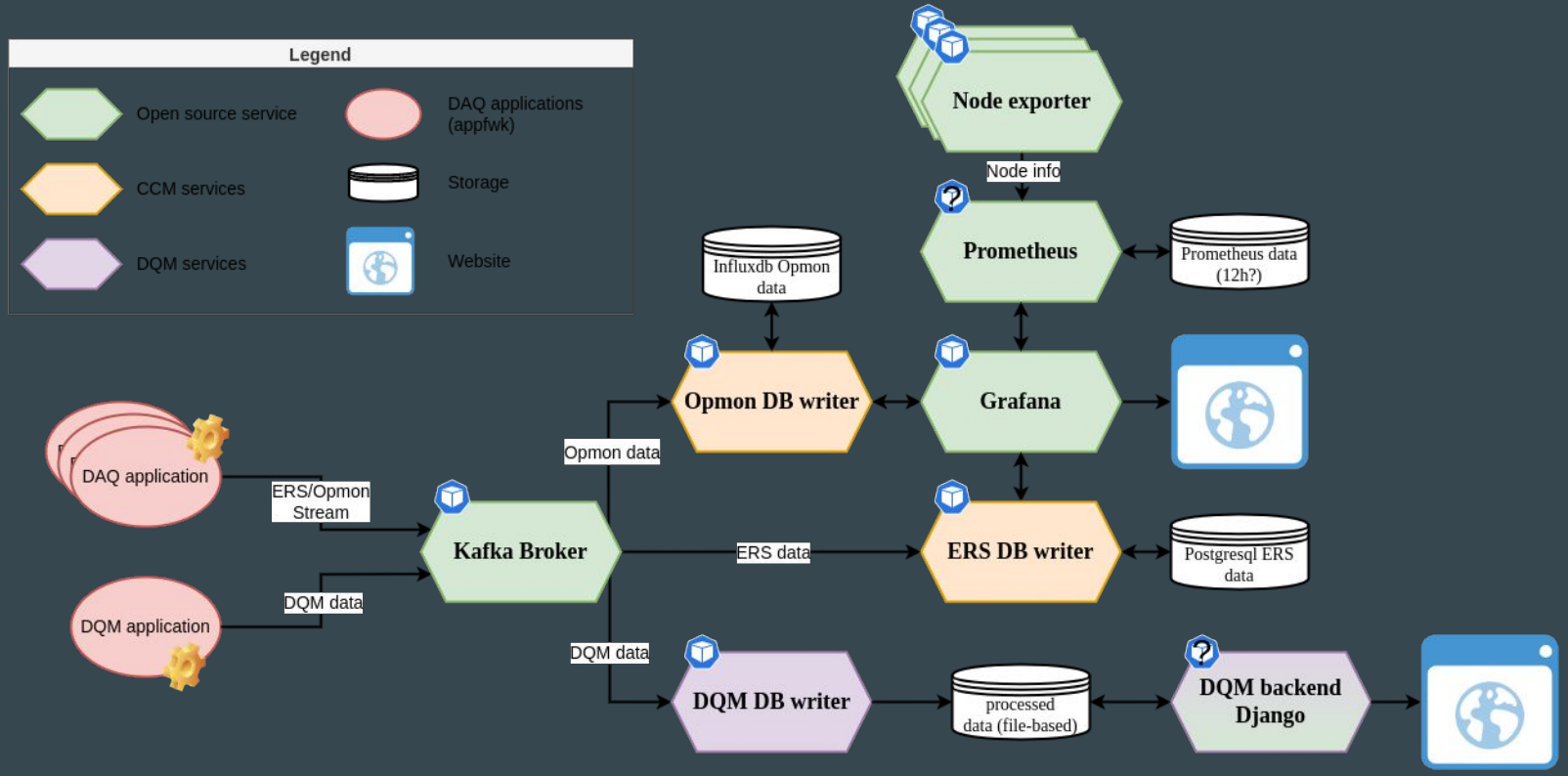
- Data streams that has to be provided to the users in real time
- For Monitoring we essentially care for two main data streams
  - Error Reporting Service (ERS) Messages
    - Code generated messages generated on particular conditions
      - info, warnings, errors
    - Structured according to a package developed in ATLAS
      - We furthered developed
      - https://github.com/DUNE-DAQ/ers
  - Operational monitoring (OpMon)
    - In simple words, metrics
    - Generated constantly so that we can understand in real time the behaviour of the system
      - And potentially take actions to correct problems
    - Structure according to DUNE specific code: https://github.com/DUNE-DAQ/opmonlib
- Every component (developers) of our system defines the information to be published
  - We take care of propagating it, store it and make it visible to the operator

The monitoring infrastructure - dummy usage

- The simplest and default usage is to print the streams in files
  - ERS messages are going in the logs
  - Opmon structured in json are out in application specific files

- Intended for testing environments
  - Specifically automated integration tests
  - Not for production conditions
    - Essentially because the information in those files is not propagated and it's lost at the end of the run
    - More advanced features require a relatively big supporting infrastructure

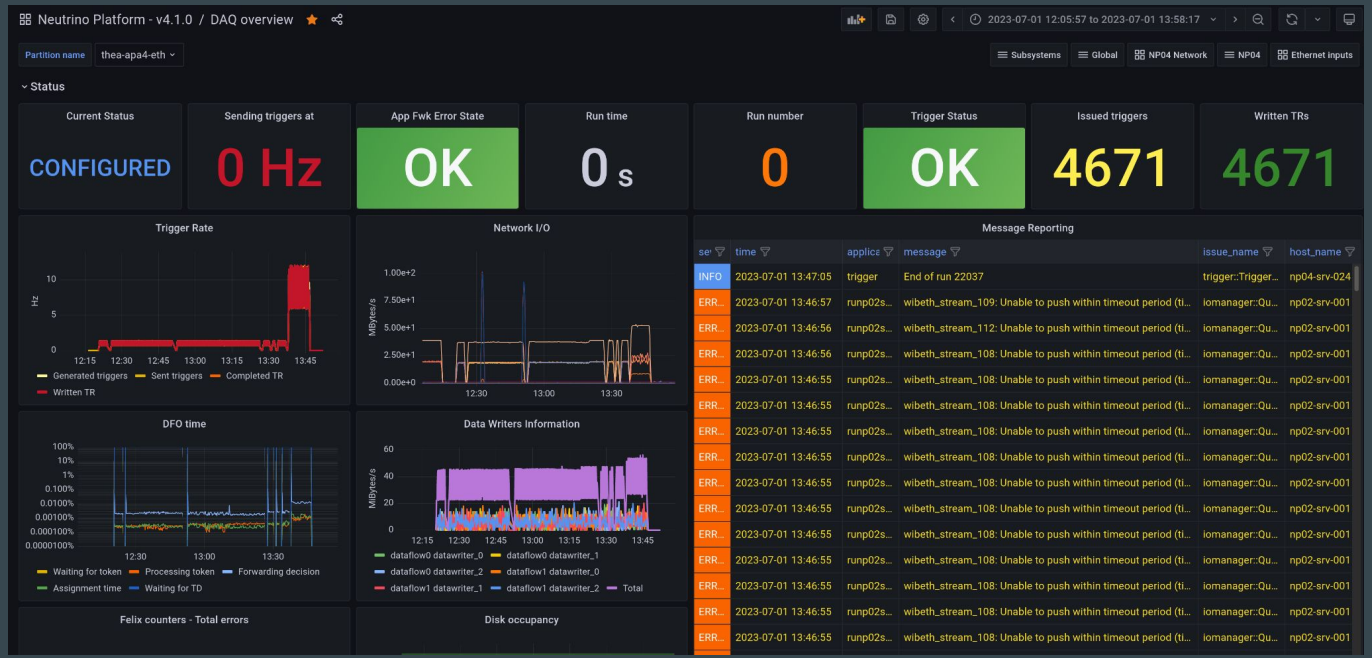
Monitoring infrastructure at EHN1



Monitoring infrastructure - Big picture

- Eventually all supporting infrastructure to live on k8s
  - Now we have around 80% in k8s
  - Goal: even in other testing facilities other than EHN1, we can easily setup everything using pocket
- ERS and Opmon info sent to a Kafka broker
  - In the form of json messages
- CCM-developed microservices read the information from kafka and put it into databases
  - InfluxDB for Opmon
  - PostgreSQL for ERS

Monitoring infrastructure - Big picture



- around 20 dashboards (and counting) to monitor different aspects of the DAQ
  - Initially drafted by experts and refined and maintained by CCM working group
  - All linked from the main one

Some details and planned improvements

- Right now messages exchanged via kafka are not strong structured
  - They are simple json strings
  - And we are not using the full potential of Kafka
    - We are essentially using only 2 topics, one for ERS and one for Opmon
    - And there are a gazillions of other options than can optimise the system
    - We are also considering alternative technologies like RubbitMQ

- In the next months we will work toward structuring and enhancing these streams
  - using Protocol Buffers schemas to serialise the messages
  - Adding more informations to keys in the messages to allow more complex functionalities
    - Like subscriptions to some specific classes of messages (source, type, ...)
  - Increasing/changing the usage of topics in the kafka broker to increase performances
    - And to make sure that we can support the stream of data we will have in the full DAQ
      - This was tested already a few months ago, borrowing the ATLAS servers

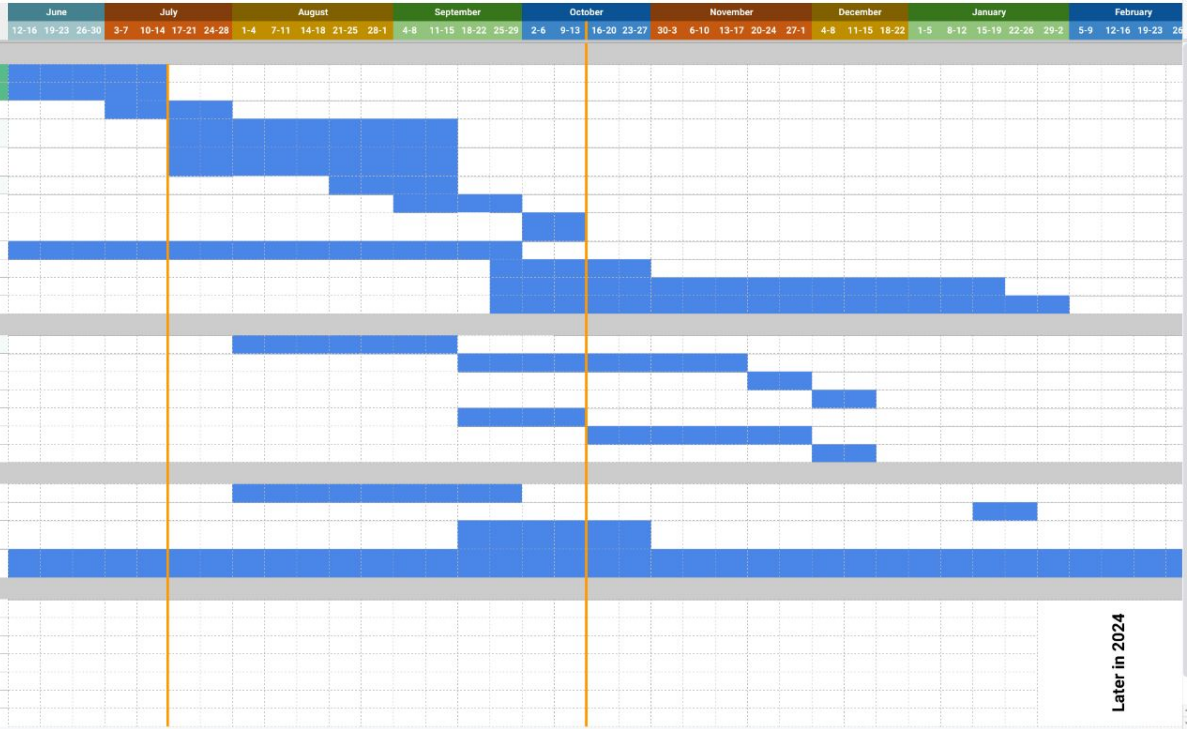
Supervisor

- A final element of the DAQ monitoring is expected to be developed
  - Based on the ATLAS and CMS experience
- System that automatically generates commands for the DAQ based on the status of the system
  - Rules to be programmed by experts
- At its core, the Supervisor is a collection of “If - then” statements
  - Performed on the information from a number of real-time sources
    - When some conditions are met
  - A dedicated interface will need to be developed between Supervisor and RunControl
  - This is an activity planned for 2024
    - After the first iteration based on ERS and Opmon only, we can also add DQM and Slow control as inputs

Gantt chart

PROJECT TITLE	Monitoring
PROJECT MANAGER	Alex, Pierre, Marco
DATE	13/06/23

WBS NUMBER	TASK TITLE	WHO?	START DATE	DUE DATE	DURATION	PCT OF TASK COMPLETE
1	**Streamer**				#REF!	
1.1	Publisher and Subscriber ERS	MR	12/05/23	14/07/2023		75%
1.2	Protobuf message schema distribution ERS	PD,JF	10/06/23	14/07/2023		75%
1.3	Update of microservices to use the ERS Streamer	MR, PD, JF,BK		14/07/2023		0%
1.4	Evaluation and improvement of ERS interfaces and ERS schema	MR/CCM				5%
1.4.1	Review and update of what is publihsed eventually on postgreSQL	CCM		29/09/2029		0%
1.5	R&D on OpMon schema and on Opmon Interface	MR/CCM				5%
1.6	Implement OpMon schema and Interfaces	MR				0%
1.7	Adapt the rest of the code to new interfaces	MR/CoreSW/CCM/NewStudent?				0%
1.8	Protobuf message schema distribution Opmon	PD,JF				0%
1.9	Presentation at CCM meeting for discussion	MR				0%
1.11	Publisher and Subscriber Opmon	MR				0%
1.12	Update of microservices to use the OpmonStreamer	MR,PD,JF,BK				0%
2	**Services**					
2.1	Broker literature review	MR/PR		15/09/2023		5%
2.2	Alternative broker R&D	MR/PR?				0%
2.3	Fake publisher developer daq_application development	MR				0%
2.4	System scaling tests including fake publisher developer	MR				0%
2.5	Review current database technology and structure	MR/?				0%
2.6	R&D on databases etc.	?				0%
2.7	Repeat system scaling tests	MR/GLM				0%
3	**Visualisation**					
3.1	Dashboard management (prod & dev instances etc.)	MR/PR	01/08/23	29/09/2023		0%
3.2	FD/ND separation	MR/ND people				0%
3.3	Review dashboard management	CCM/Infrastructure				0%
3.4	Continuous developer support	MR/NewStudend??	01/01/23			
4	**Supervisor**					
4.1	First automated action demonstration	MR/PL				0%
4.2	Write out documentation	MR				0%
4.3	Presentation at a General DAQ meeting	MR				0%
4.4	System experts to code their automated actions (CoreSW)	CoreSW				0%
4.5	System experts to code their automated actions (Timingh)	Timing				0%
4.6	System experts to code their automated actions (Trigger)	Trigger				0%
4.7	Etc.	All subgroups				0%



Take away messages

- Busy schedule for the next 6 months
  - And that is mainly for requirements for the supervisor
  - Then we can move forward to the supervisor
- All the preliminary R&D conducted so far is encouraging
  - We proved that the design work
  - We can support the monitoring data streams at a DUNE scale
- The current system is widely used at EHN1
  - With constant monitoring features continuously added