

# (Proto)DUNE Prompt- Processing Infrastructure

Lino Gerlach<sup>1</sup>, Torre Wenaus<sup>1</sup>, Xin Qian<sup>1</sup>, Brett Viren<sup>1</sup>, Michael Kirby<sup>1</sup>

<sup>1</sup>Brookhaven National Lab (US)

10<sup>th</sup> April 2024

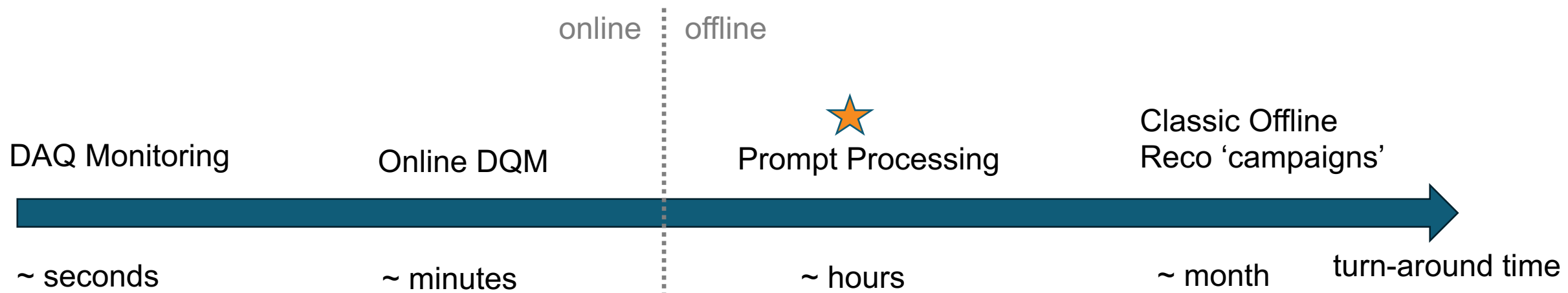
# Overview

- Before leaving DUNE, I did some research on infrastructure for Prompt-Processing
  - Today: Overview of what I found & where it stands
- Required features of a prompt-processing system
- What (partial) implementations exist?
  - Local demonstrator prototype
  - Used in Run 1: p3s (from Maxim Potekhin)
  - A possible solution based on POMS

# Prompt Processing - Intro

What do I mean when saying 'Prompt Processing'?

- Offline data analysis as it comes in ('promptly')
- Run first steps of full reco so that output can be used for further analysis
- Record data quality metrics on the side (DQM)
- Distinguish between payload and infrastructure (focus of this talk)





# Online DQM vs Prompt Processing

## Online DQM

- Fixed computing resources
- Consider fraction of data
- Output only high-level metrics
- Run light-weight analysis (e.g. calculate RMS, tracking w/ LARDON)
- Conservative w.r.t. to changes to code



## Prompt Processing

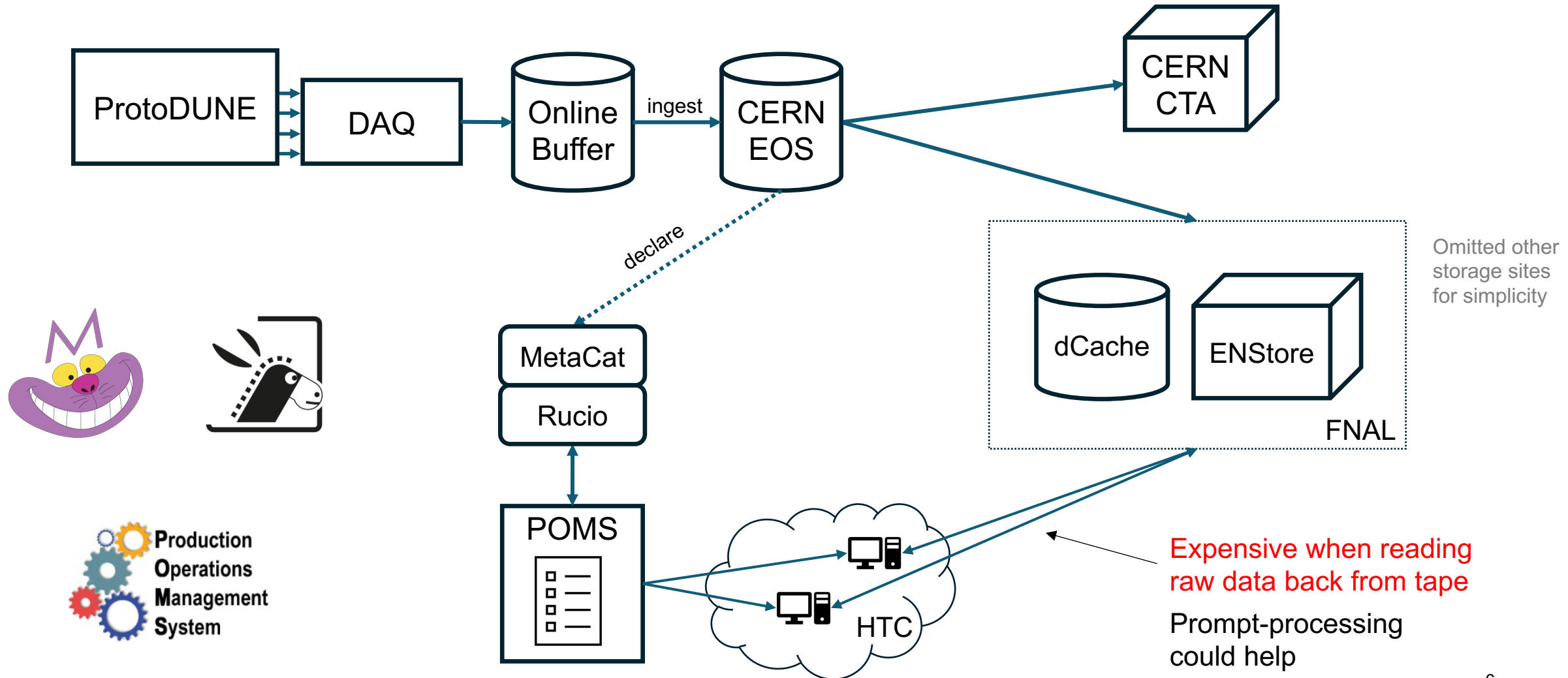
- Flexible computing resources
- Consider all incoming data
- Output high-level metrics & resulting data products
- Run first steps of full offline reco ((partial) LArSoft or WireCell job, ...)
- Very flexible w.r.t. to changes to code



# Advantages of Prompt-Processing

- Running complex analysis promptly: spot potential problems earlier
  - Problems in raw data overlooked by online DQM
  - Problems in unseen fractions of raw data
  - Potential bugs in offline reco software
- Minimize fraction of raw data read-backs from tape
  - Not a problem for ProtoDUNE, but for other prototypes and DUNE

# ProtoDUNE II Data Pipeline - Overview

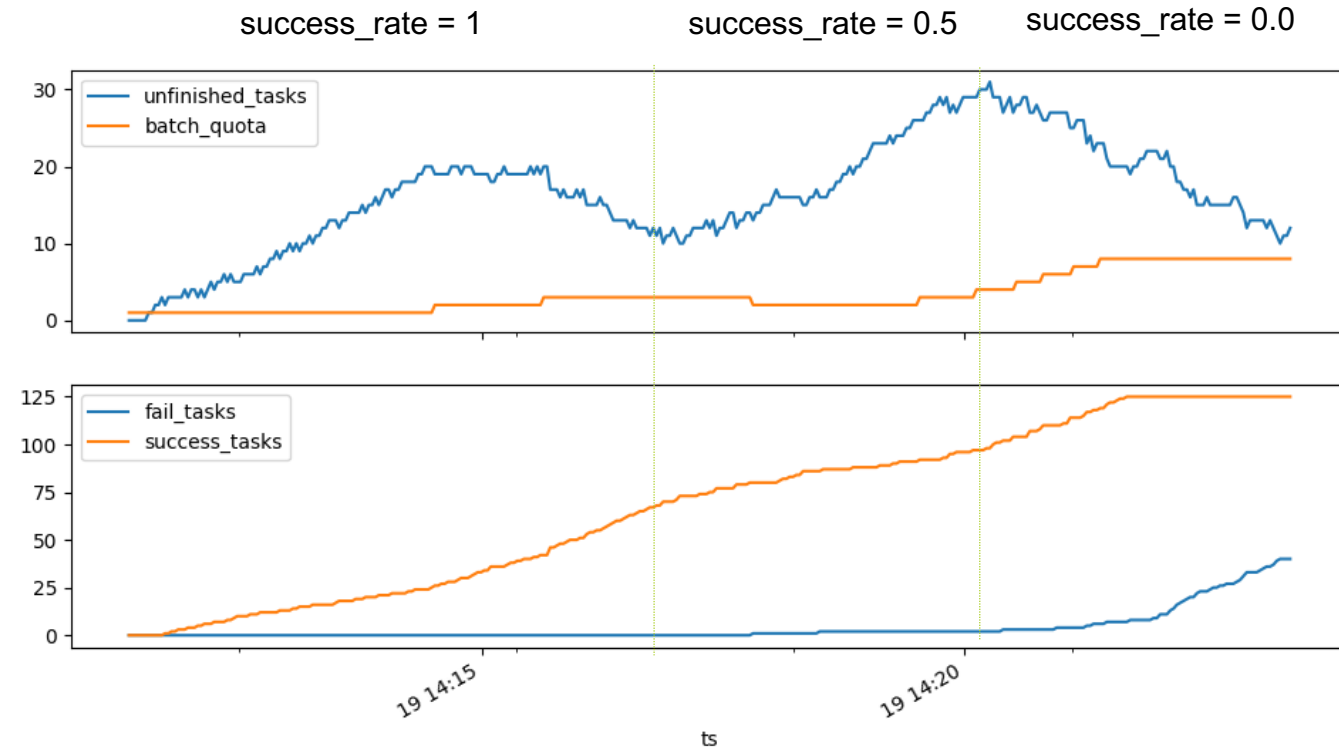


# Prompt Processing – Needed Features

- Automatically detect new raw data in buffer
- Launch (parallel) jobs with same payload (NF & SP?)
  - Number of parallel jobs depending on backlog
- Record which subset of data was successfully processed
  - Possibly restart failed jobs n times
  - Mark repeatedly failing jobs for reprocessing later
- Keep data to-be-processed on disk buffer for as long as possible
  - Move successfully processed files from disk buffer
- Provide monitoring dashboard
  - Number of jobs currently running, fraction of successfully processed data

# Local Prototype Implementation (Demonstrator)

- Partial Prototype implementation (python): <https://github.com/ligerlac/prompt-processing>
  - Separates **file handling**, **batch handling**, **book keeping**
  - Separates interface + implementation
    - Easy to replace batch system backend (or book keeping DB)
- Can be run as single python script or independent cron job(s)





# What had been done in past

An offline DQM system was used for ProtoDUNE SP:

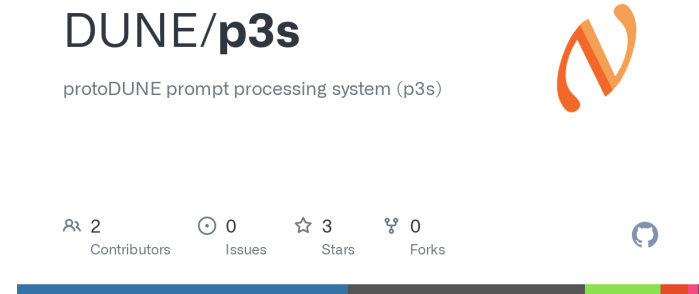
- ‘ProtoDUNE Prompt Processing (p3s)’ by Maxim Potekhin (<https://github.com/DUNE/p3s>)

## Purpose

- Continuous low-latency processing of data for data quality monitoring
  - Slower than online monitoring, faster than full processing
- Easier to deploy than more complex systems (e.g. PanDA) w/ subset of functionality

## Features

- User-defined data reconstruction jobs
- Pilot-based system to avoid slow batch response
- Flexible usage of computing resources



DUNE/p3s  
protoDUNE prompt processing system (p3s)

2 Contributors 0 Issues 3 Stars 0 Forks

# p3s - Existing Solution?

## Problems with p3s

- Original developer left DUNE (not maintained anymore)
- Only consider (user specified) fraction of data
- Only output high-level metrics – not the resulting data products

p3s could probably be adapted to our needs, but:

- Ideally, leverage existing DUNE tools
  - E.g. workflow management (POMS), data movement (Rucio)
- Still some open questions:
  - How sluggish is grid submission & data movement?
  - How much computing resources are needed?

# Proposed Implementation

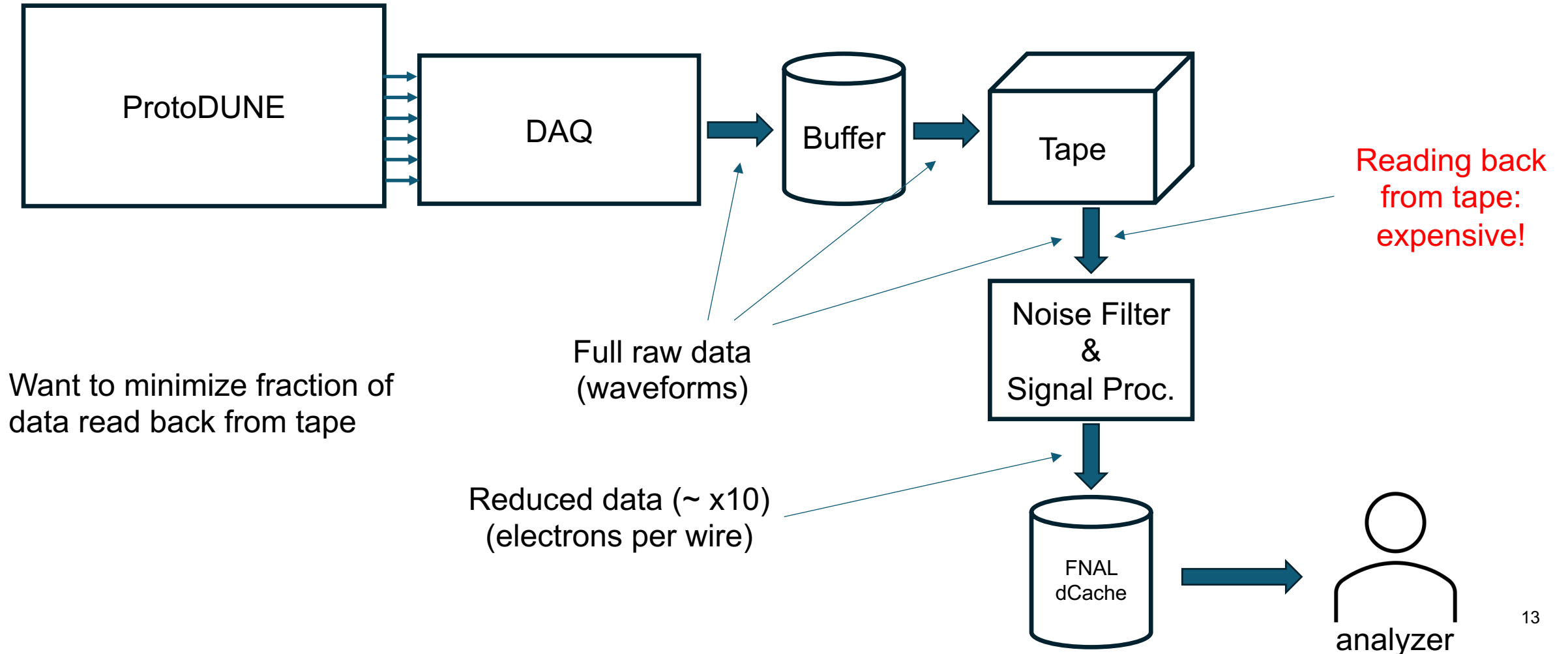


- Single POMS campaign restricted to run @ CERN
  - Use POMS' submission scheduler ('cron' - like 'Draining Dataset' from Computing tutorial)
    - <https://dune.github.io/computing-training-basics-short/08-poms-part2/index.html>
    - 'Drain' raw dataset declared in sam by 'declare daemon'
  - Multi-stage workflow:
    - 1. stage: NF & SP
    - (2.) declare resulting DS in Rucio, Metacat and SAM
    - 3. perf. Evaluation
    - 4. Upload perf. Evaluation results to DQM dashboard (see Gabriela's talk)
- Integrated functionality only for SAM
  - But: can run any bash script -> Rucio & Metacat already possible
  - Rucio integration already on the way (POMS dev. instance w/ Rucio)

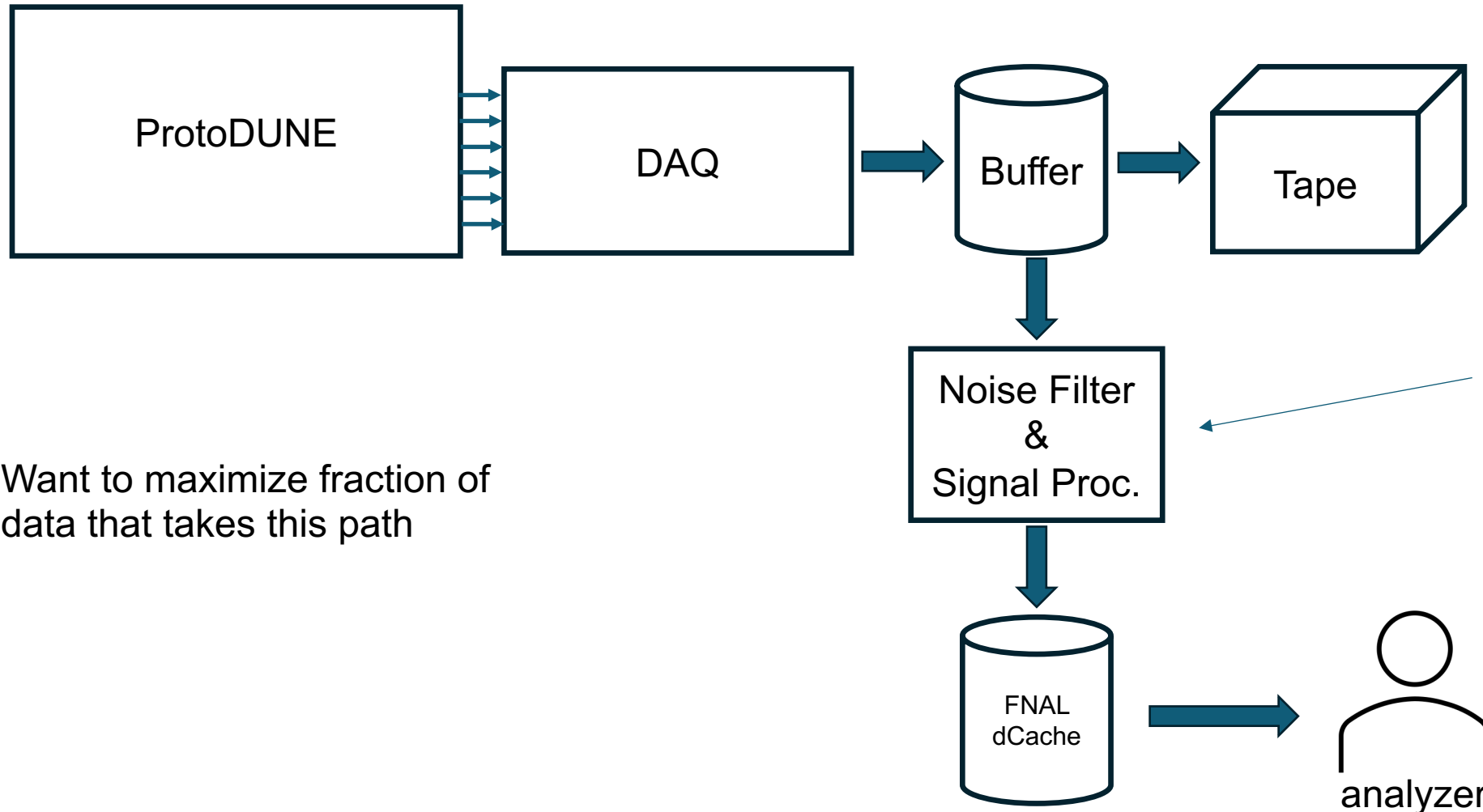


# Backup

# Simplified Data Pipeline – Worst Case



# Simplified Data Pipeline – Best Case



Want to maximize fraction of data that takes this path

happens within lifetime of buffer: **prompt**