Monitoring Data Mover Latency and Transfer Rate

SLAC NATIONAL ACCELERATOR LABORATORY

Joshua Matni

LCLS Intern in TID CDS Advanced Data Systems under the PSDM team

U.S. DEPARTMENT OF ENERGY Office of Science

Introduction

The LCLS datamover is the pivotal element responsible for the transfer of experiment data files to various storage resources. The **transfer speed** is crucial, depending on the specific storage resource in use. This rapid data transfer is instrumental in enabling users to efficiently process and monitor data quality, which subsequently informs decisions concerning the continuing data collection process.

The data movement mechanisms are illustrated by dashed and solid arrows, representing writes from the Data Acquisition (DAQ) and data reduction pipeline (DRP, LCLS-II), and the activities controlled by the data movers, respectively. There are two types of transfers: 1) **Real-time transfers**, which begin as soon as a file is opened, with the mover concurrently reading data while the DAQ/DRP writer is still engaged in writing; 2) **Complete or closed file transfers** that only commence after a file has been closed.

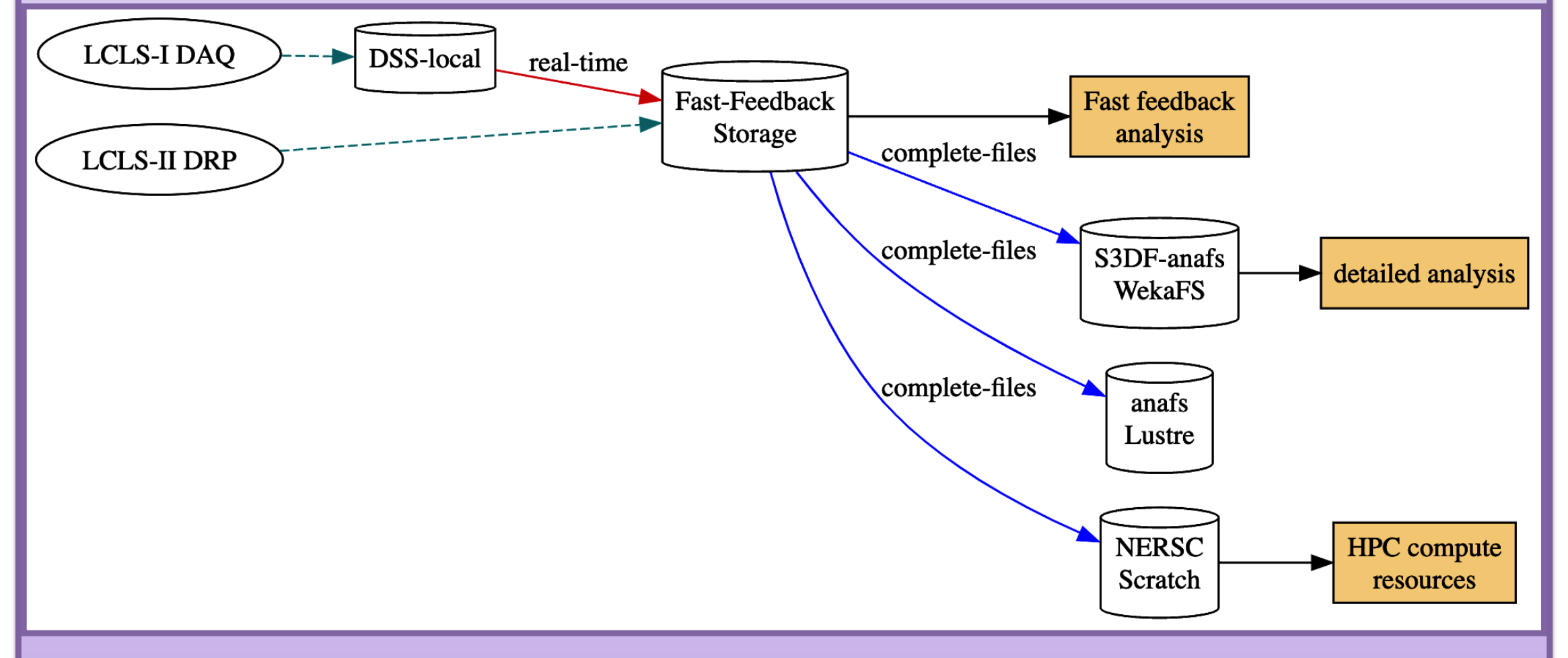
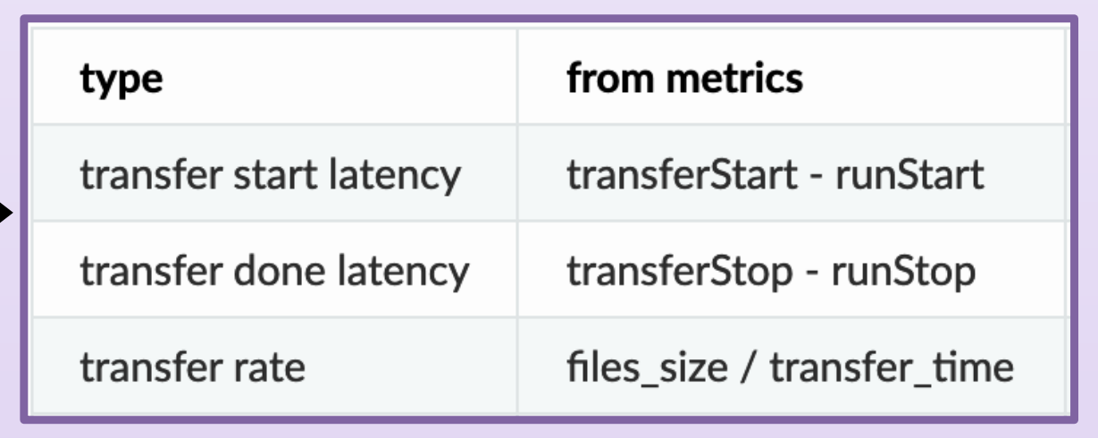


Fig. 1: Flow of Data from LCLS I & II, Courtesy of Wilko Kroeger

Goals

To show how quickly files/runs are transferred between the different storage resources

Values we are interested in



type	from metrics
transfer start latency	transferStart - runStart
transfer done latency	transferStop - runStop
transfer rate	files_size / transfer_time

- ✦**elog_runs metrics:**
  - Run #, start and stop time of a run
- ✦**elog_file_catalog metrics:**
  - Name, path, run #, create_time and location (file_size, only valid if file is transferred)
- ✦**moverDB transfers (file_migration) metrics:**
  - Transfer start and stop times, file path, and source

Where are these metrics stored?

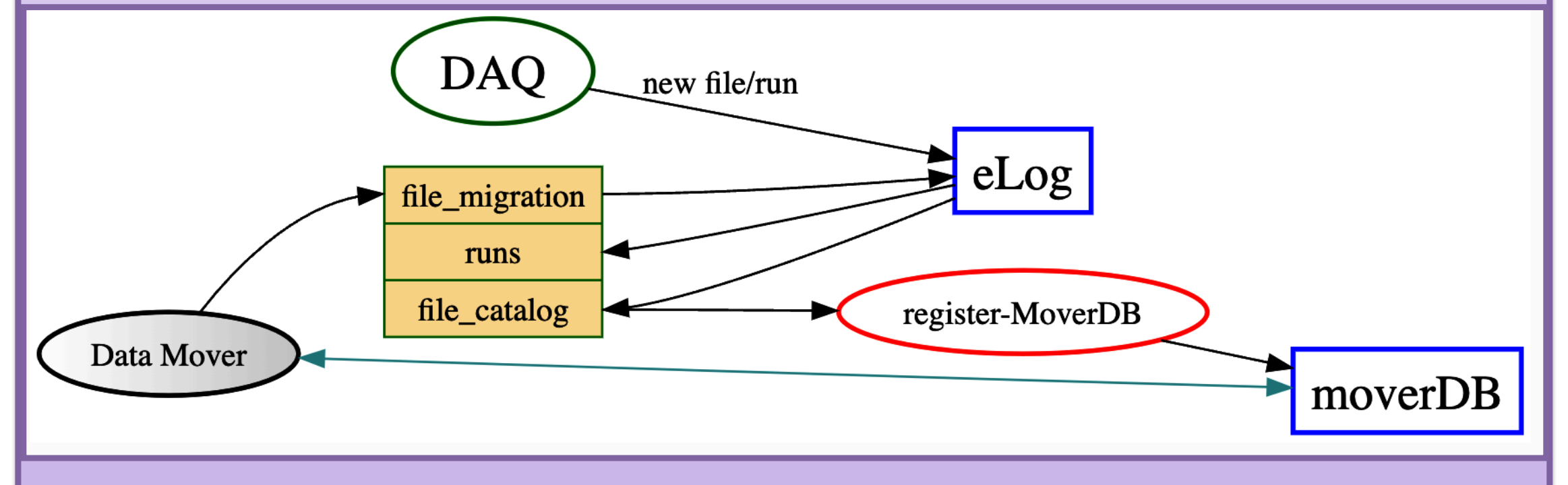
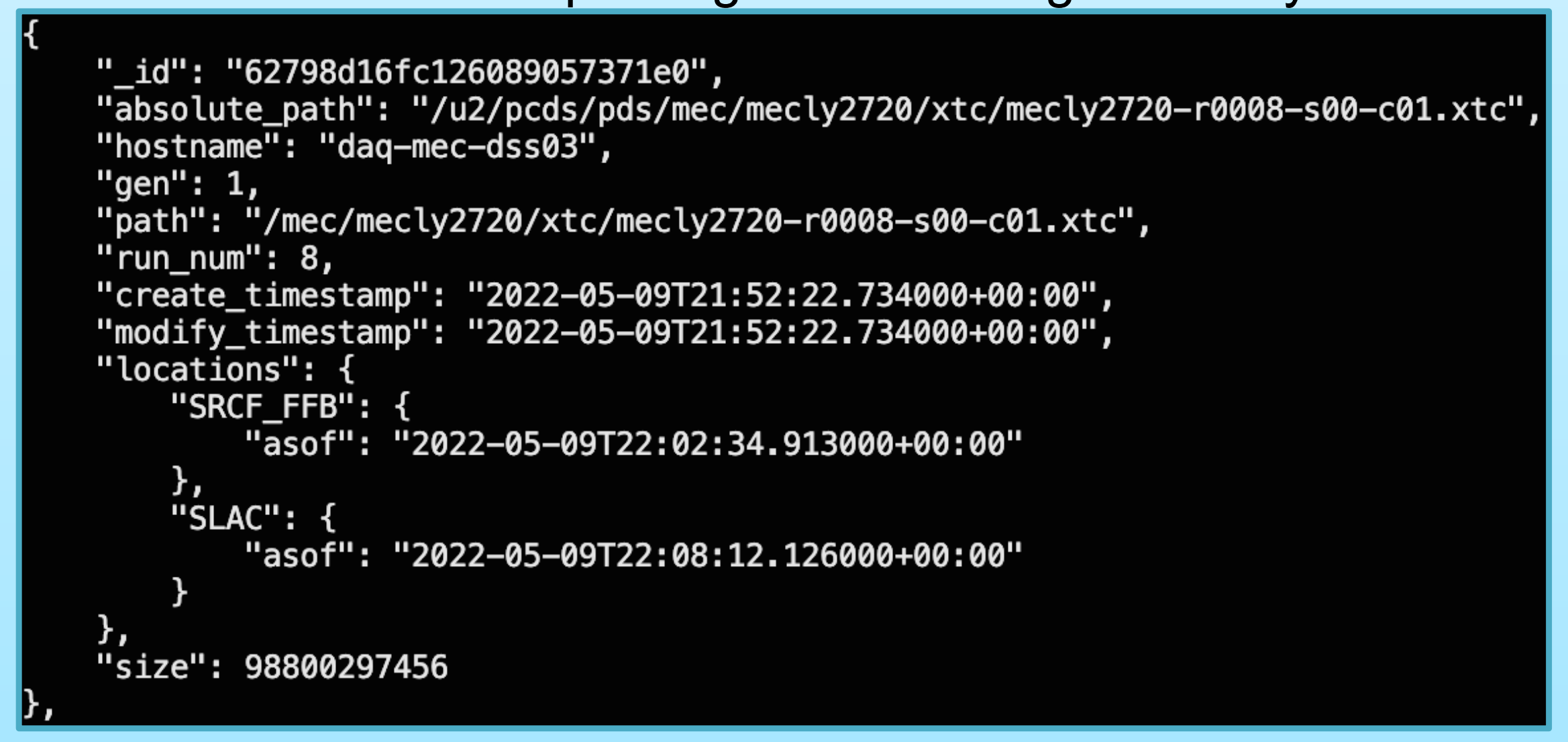


Fig. 2: Flow of messages between the storage (eLog, moverDB) and the kafka topics (tan boxes), Courtesy of Wilko Kroeger

Objectives

Get All Files/Runs of a LCLS Experiment:

- Each approved experiment proposal receives a set beam time, typically five 12-hour shifts. Within these shifts, hundreds to thousands of runs are performed, with varying configurations and tasks.
- Experiments originate within LCLS-I (cxi, mec, mfx, xcs, xpp, det), and LCLS-II (tmo, txi, rix). We obtain them by operating the python requests library to gain API access to the LCLS-II elog **(Fig. 3)**, and executing SQL queries for gathering the moverDB file transfers. We then utilize JSON parsing for formatting and easy extraction.



```
{
    "_id": "62798d16fc126089057371e0",
    "absolute_path": "/u2/pcds/pds/mec/mecly2720/xtc/mecly2720-r0008-s00-c01.xtc",
    "hostname": "daq-mec-dss03",
    "gen": 1,
    "path": "/mec/mecly2720/xtc/mecly2720-r0008-s00-c01.xtc",
    "run_num": 8,
    "create_timestamp": "2022-05-09T21:52:22.734000+00:00",
    "modify_timestamp": "2022-05-09T21:52:22.734000+00:00",
    "locations": {
        "SRCF_FFB": {
            "asof": "2022-05-09T22:02:34.913000+00:00"
        },
        "SLAC": {
            "asof": "2022-05-09T22:08:12.126000+00:00"
        }
    },
    "size": 98800297456
},
```

Fig. 3: A file from the elog_file_catalog; experiment: mecly2720

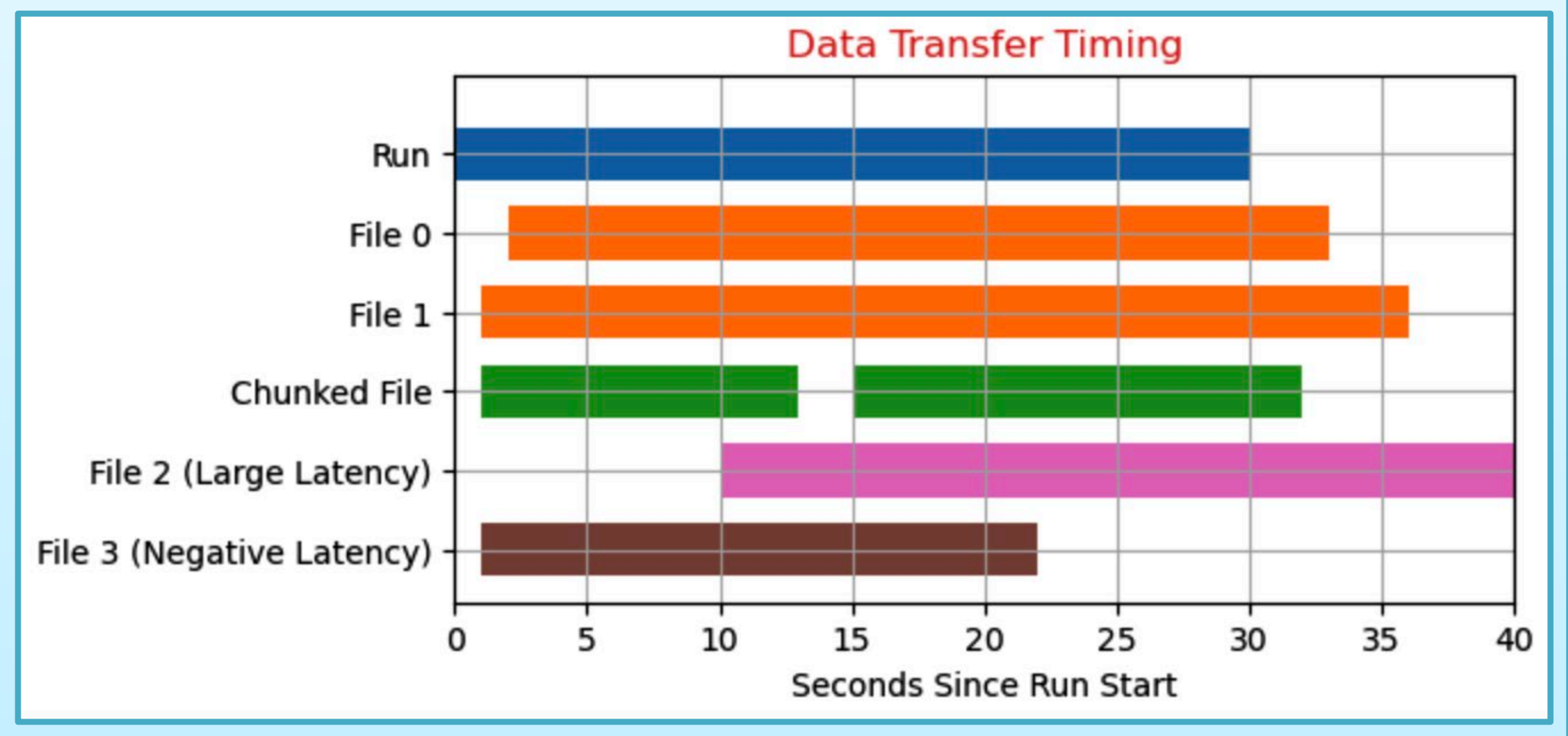


Fig. 4: Example of metrics timings

Metrics Considerations:

- Files/Runs

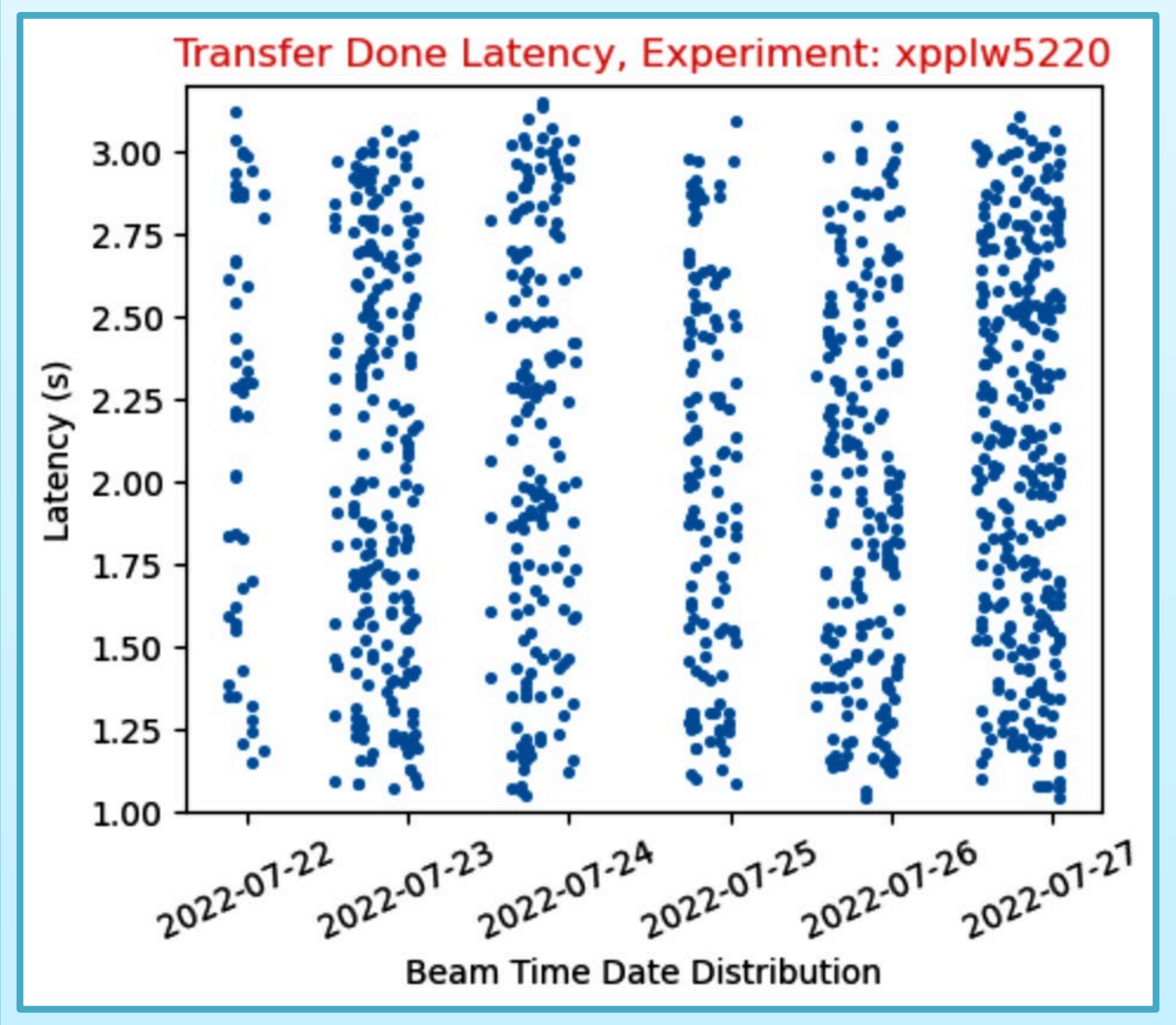
A single run consists of many files being created and written in parallel. Therefore for the metrics, it is not only important to look at files, but also when all files of a run have been transferred.

- Chunked Files

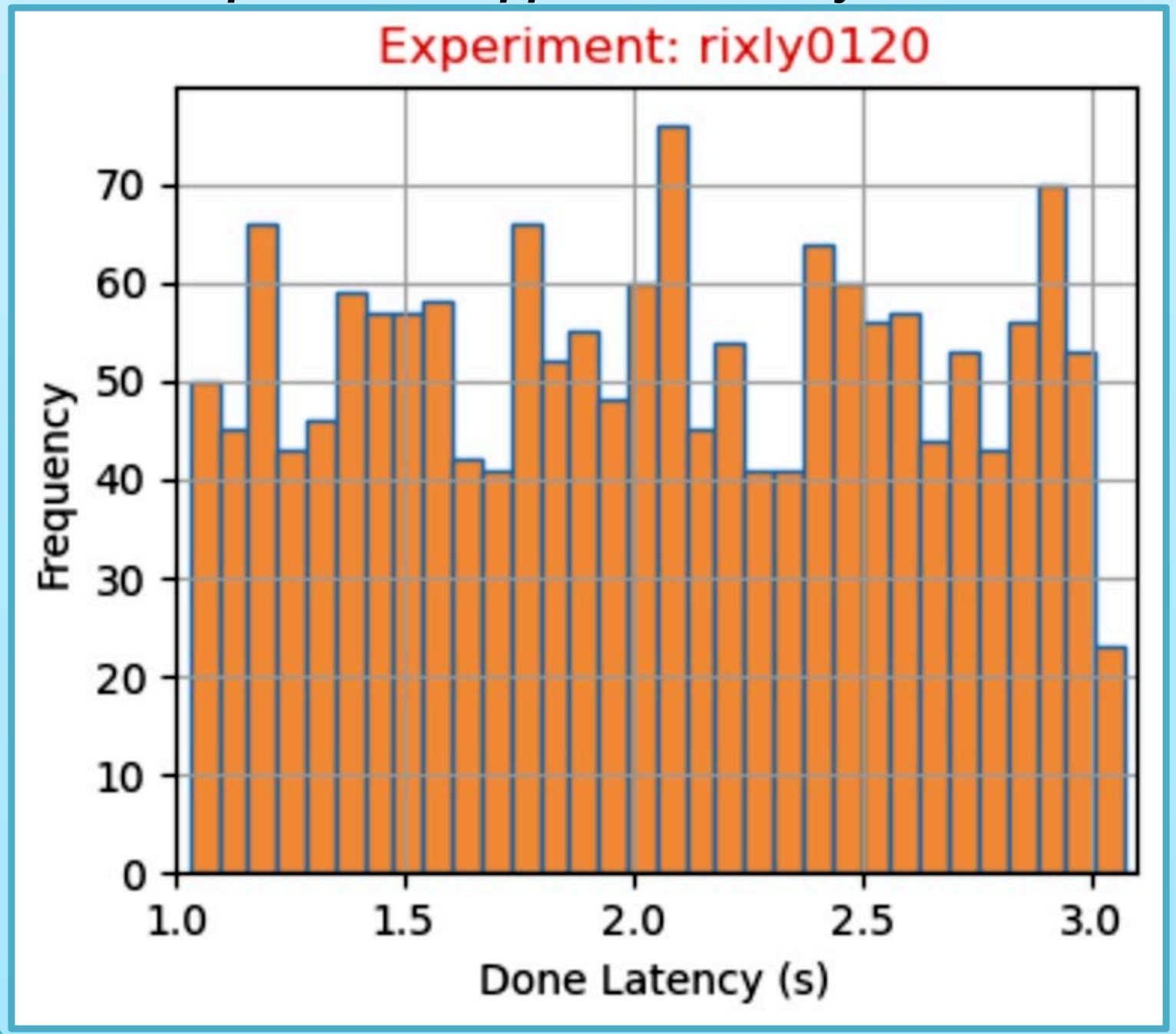
A max file size is enforced by the DAQ. If a file reaches the max it will be closed and a new one is opened with the same name except the chunk is incremented by one. Different chunks are used to calculate the start and done latencies.

Analysis of Latencies

Expected Latencies

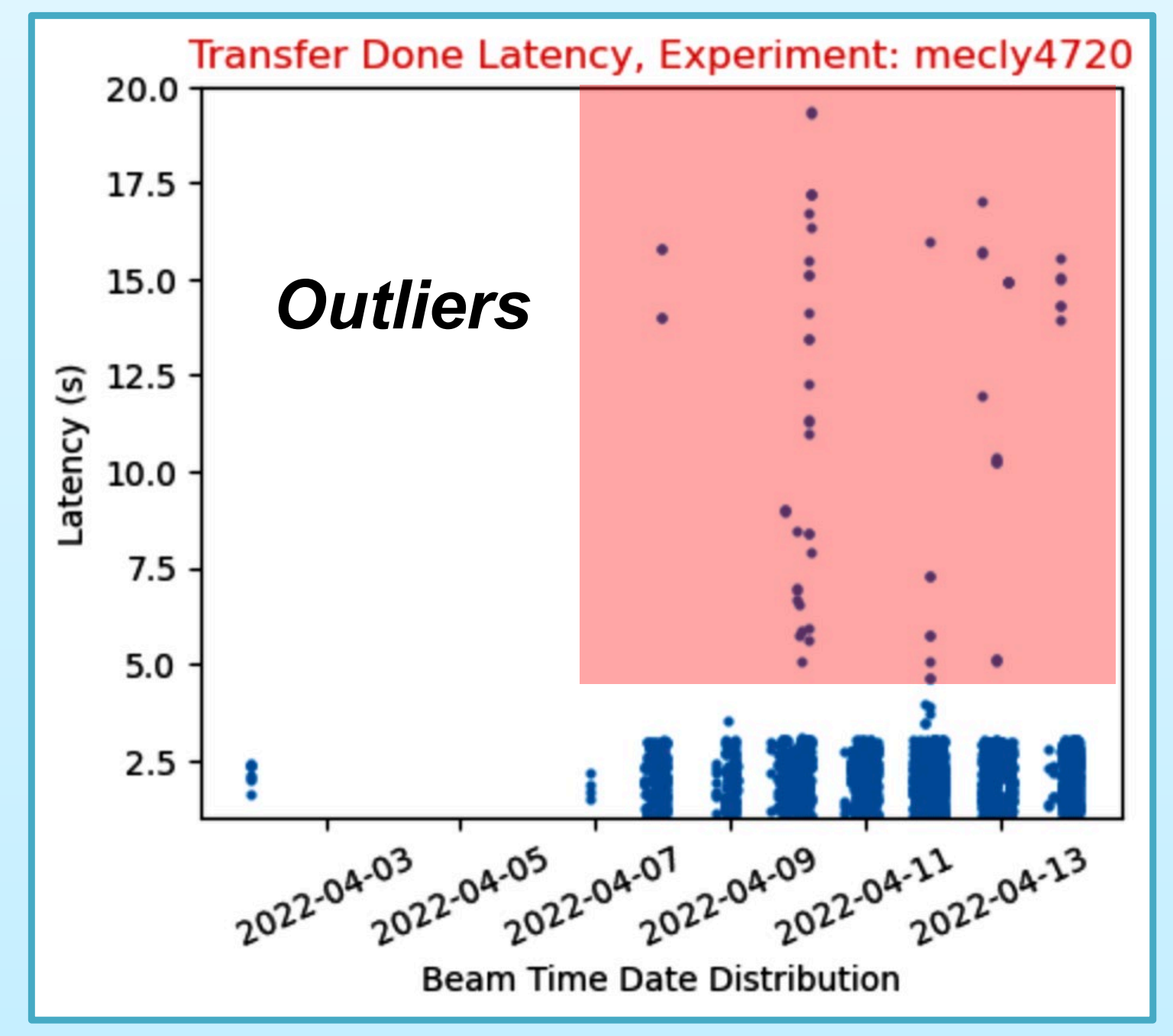


Figs. 5 & 6: Distribution of done latency of experiments: xpplw5220 & rixly0120

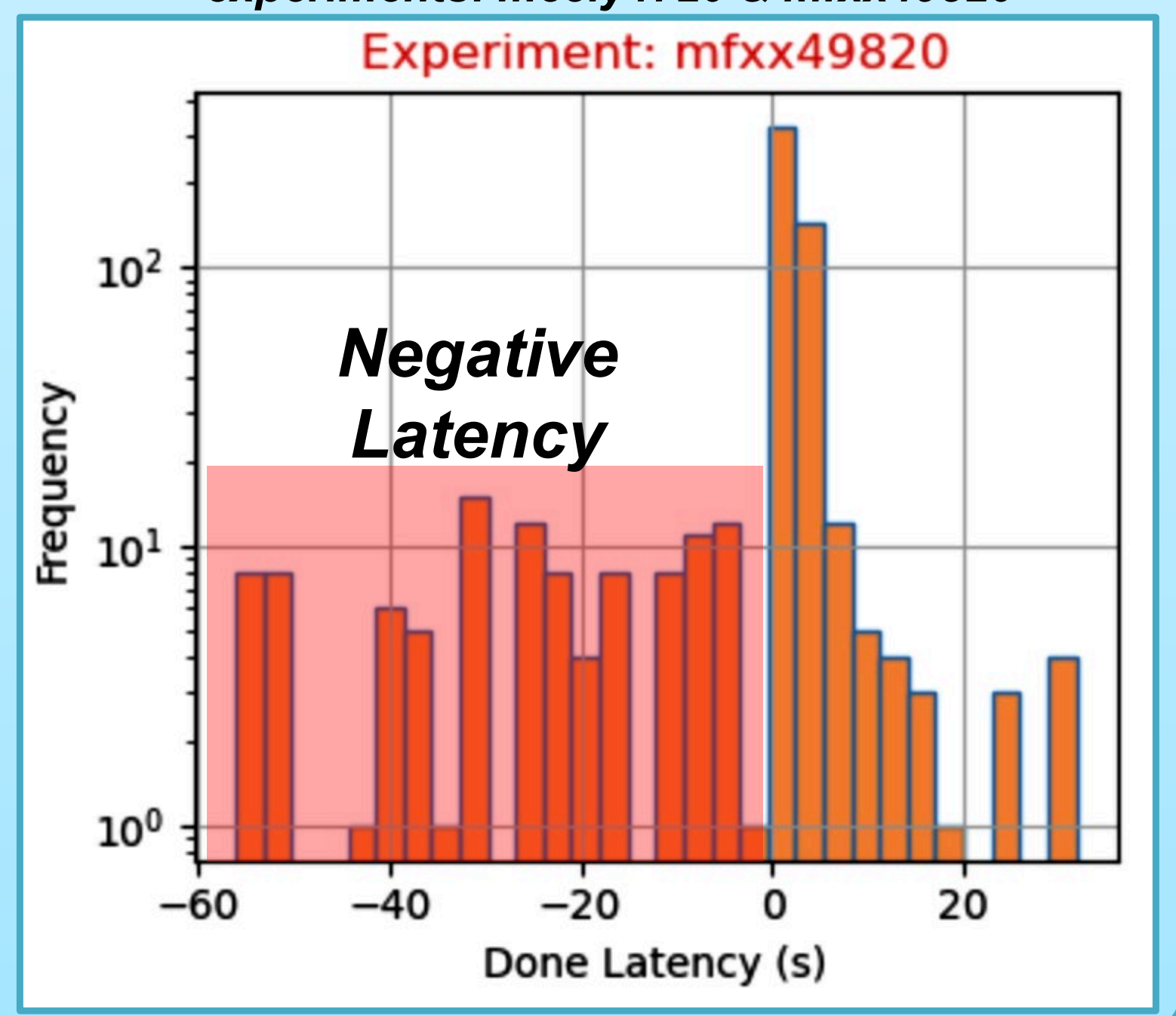


- After retrieving and processing all experiments, we organize them into a data frame using pandas. We then analyze our metrics with Jupyter and visualize them with Matplotlib.
- **(Figs. 5 & 6):** We expect the distribution of the done latency to be around **2-3 seconds**.
- **(Fig. 7 & 8):** However, several experiments have outliers that exceed our expected range. Potential reasons for this include:
  - Large start latencies.
  - Chunked files.
  - Complications with storage resources: issues with the DSS node or DAQ transferring data or even crashing. This explains the negative values.
  - Files may transfer before a run ends; review all files in the same stream.
- LCLS Data movers are dependent on the completion of a previous transfer, any delay in this process directly impacts the total execution time. Therefore, **optimizing done latency** is crucial in improving the overall performance and efficiency of our system.

Unusual Latencies



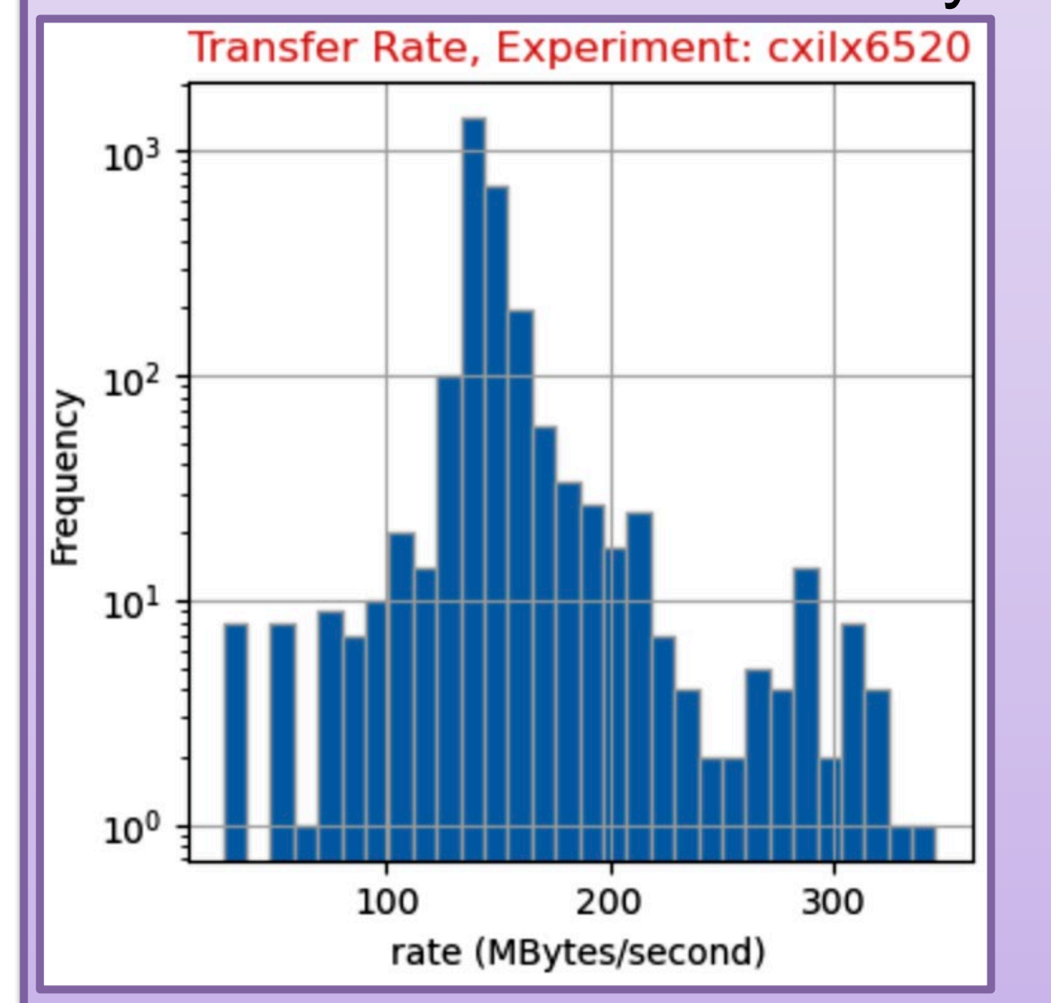
Figs. 7 & 8: Distribution of done latency of experiments: mecly4720 & mfxx49820



Transfer Rates

Transfer rates vary and depend on the experiment; however, there are two limitations:

- **(Fig. 9):** Real time transfers are limited by data collection rate which could be around 100-300 MB/s (or slower).
- **(Fig. 10):** If a transfer starts very late (large start latency) the rate will be limited by the checksum calculation of



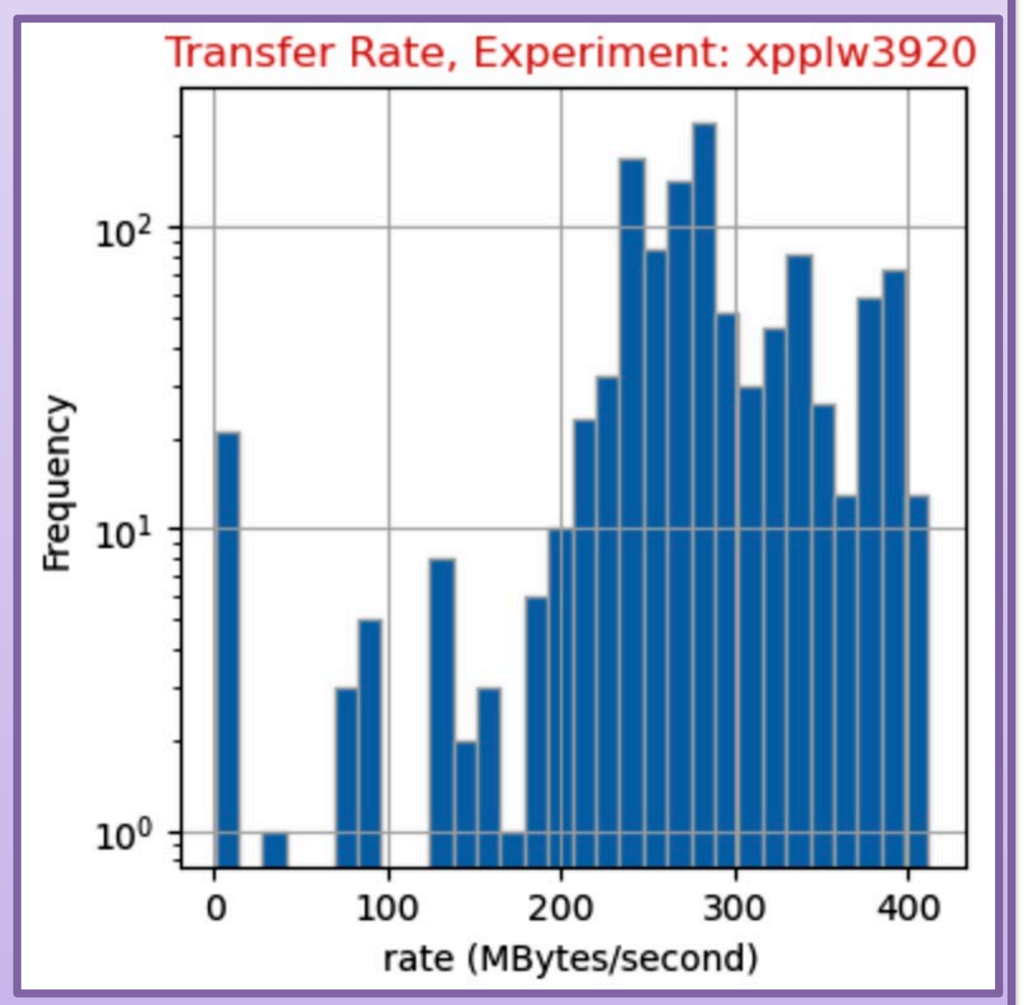


Fig. 9: Experiment cxilx6520 *Fig. 10: Experiment xpplw3920*

Conclusions and Continuations

Beam users rely on rapid feedback loops where each run's data is analyzed, adjustments are made and subsequent runs are planned. We have shown getting this feedback within a few seconds relative to the data collection allows to optimize the desired physics results ensuring an efficient usage of the precious LCLS beam time. However, occasional outliers were identified.

Monitoring data mover latency and transfer rates are equally important for a few reasons. Firstly, they are vital indicators of the system's performance and health. Any abnormality, like increased latency or decreased transfer rate, could signify potential system issues, including network congestion or disk failures. Secondly, the expensive nature of the operations makes it essential to maintain optimal efficiency. By monitoring these metrics, one can spot inefficiencies, helping in cost management.

Looking towards the future, we've initiated a process referred to as "**Investigative alerting**," which is designed to boost our responsiveness to potential issues. This system sends out automatic messages to the PSDM team if any suspicious activities or anomalies are detected in data transfers or ongoing experiments. The automated alerting acts like an alarm, capturing our immediate attention and enabling swift actions to rectify any issues.

Acknowledgments

Thank you to the **Photon Science Data Management** (PSDM) team: **Murali Shankar** and **Wilko Kroeger** for the unwavering support and guidance, and **Thorsten Schwander** for the invaluable advice on logical thinking, and modularity that played a key role in this project's development.