ASAPO - A high-speed streaming framework to support an automated data-processing pipeline.

Mikhail Karnevskiy

SciComp Workshop

1 July, 2024

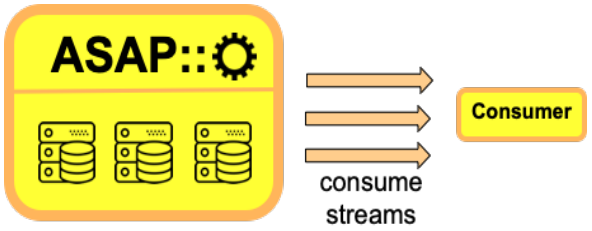
ASAP::O introduction

ASAPO is a framework to transfer data between different data-source, data-processing and data-storage components.

Key points:

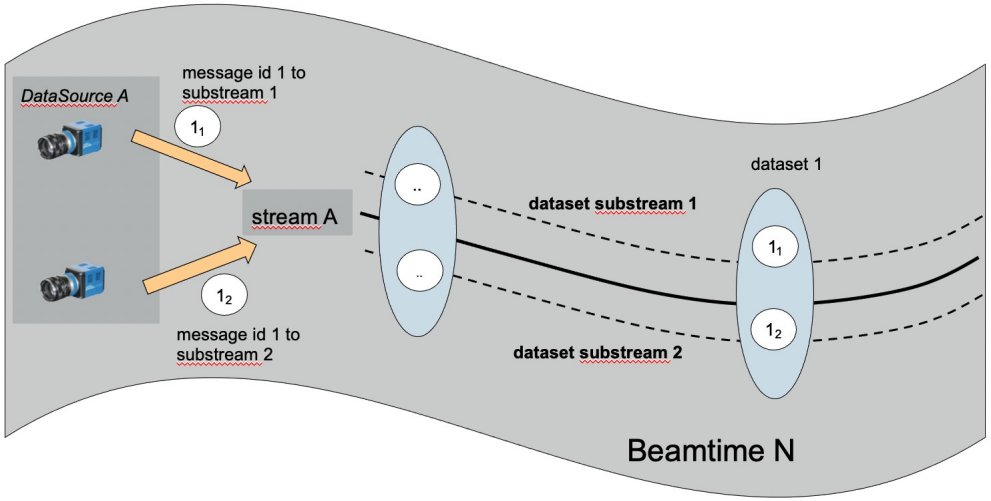
- High-bandwidth communication between state-of-the-art detectors, the storage system, and independent analysis processes across DESY facility
- In-memory data transfer with optional caching on disc
- Easy to use C++/Python interfaces:
  - Producer: sends data to ASAPO
  - Consumer: get data from ASAPO
- Data in ASAPO is organized in tree structure:

  Beamtime (experiment name)
  Data-source
  Stream
  Message



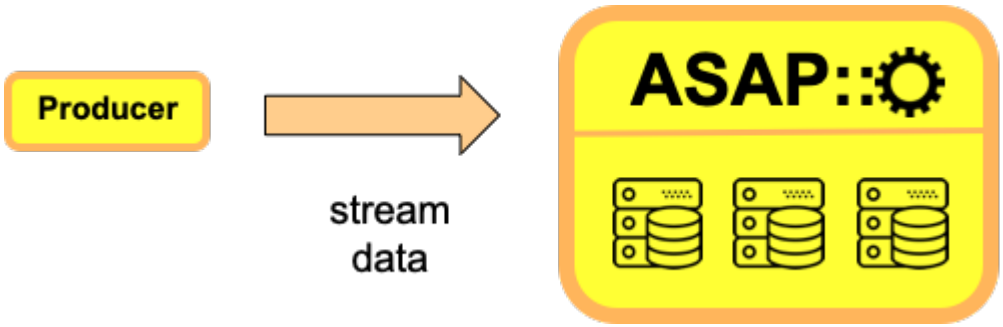
Data in ASAPO

- Messages are uniquely identified by beamtime, data-source name, stream name, and index
- Data-source name and stream names are arbitrary strings defined on client side
- Messages are automatically indexes from 1 to N. User can give a user-index.
- Each message contains a binary data blob and a JSON metadata.
- Data is stored in memory-cache and on disk (optionally), metadata is stored in database
- Additional beamtime and stream metadata documents
- Several data-sources can be combined in asapo to a dataset



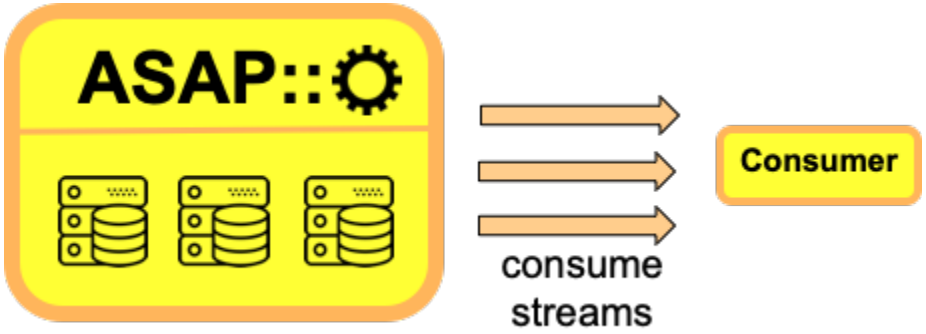
Ingest data

- Ingest to a specific data source and stream
- Messages are indexed automatically. (Optionally indexed by user)
- Multiple producers can send to the same stream in parallel
- Transfer acknowledged when completed (stored in a filesystem, and recorded in database)
- Automatic re-transfers in case of failure (until configurable timeout)
- Non-blocking with callback
- Synchronization between data-sources is achieved by combining them in asapo dataset

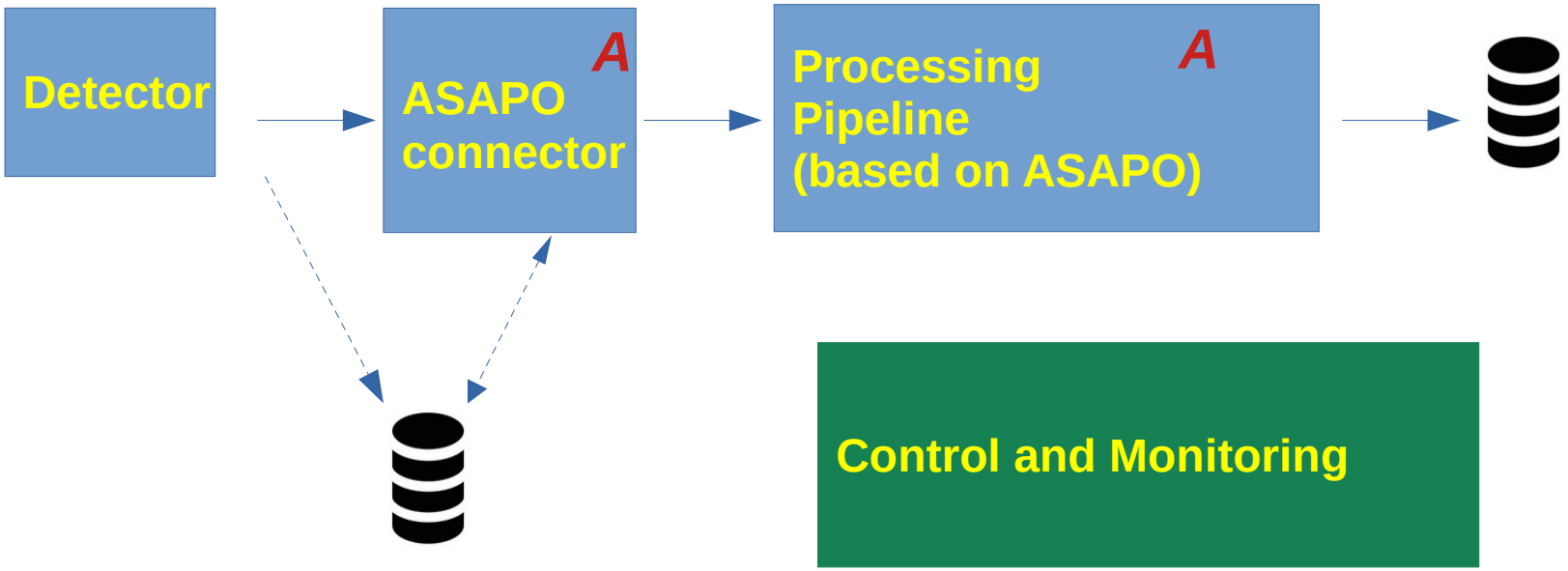


Retrieve data

- Read data/metadata message by message from a specific data source and stream
- if data is not in buffer, it will be read from file-system (Same interface for online and offline)
- Optional offline data transfer for consumers without file-system access
- Calls block until message is received (or timeout occurs)
- Multiple access modes:
  - I *get_next(group_id, ...)* for parallel online data processing, consumers sharing the same group_id receive different messages. Resend message in case of failure.
  - I *get_last* for online visualization
  - I *get_by_id(index, ...)* for random access



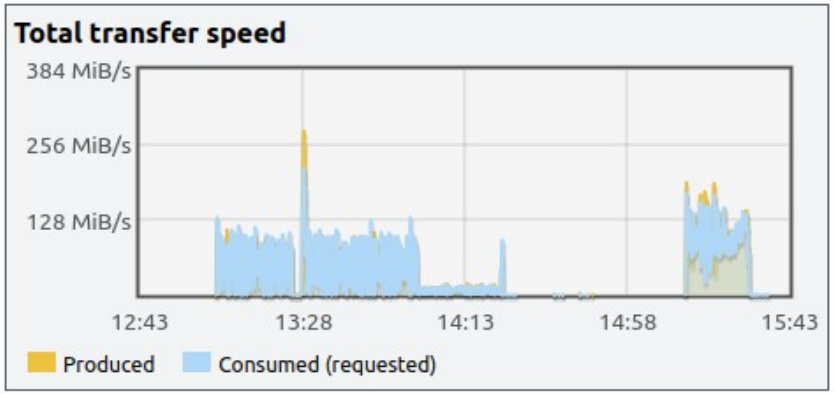
Data flow with ASAPO

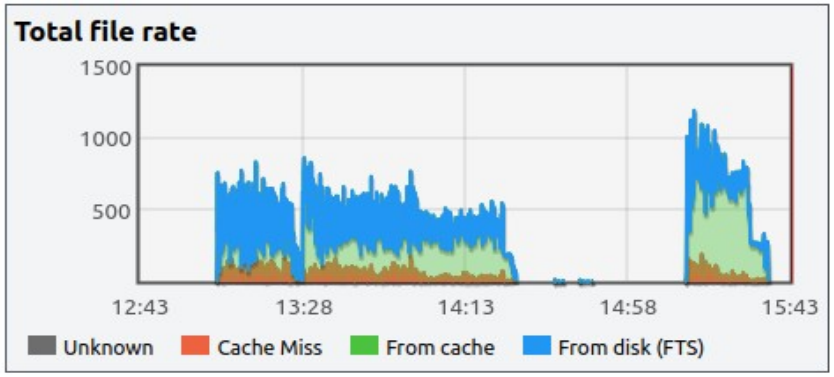


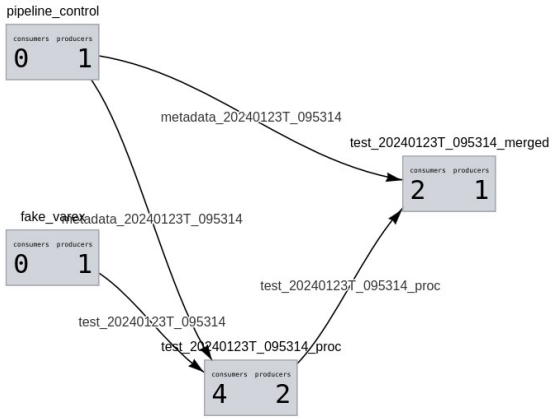
- ASAPO-connector is detector specific part
- Saving of Raw data may become optional
- Control and monitoring is decoupled from the pipeline and have a GUI.
- **A** – based on ASAPO service

Monitoring

- Web service to visualize data-flow is ASAPO
  - Show message rate and file rate
  - Show delay in data processing
  - Visualize the pipeline topology
- Further development is required

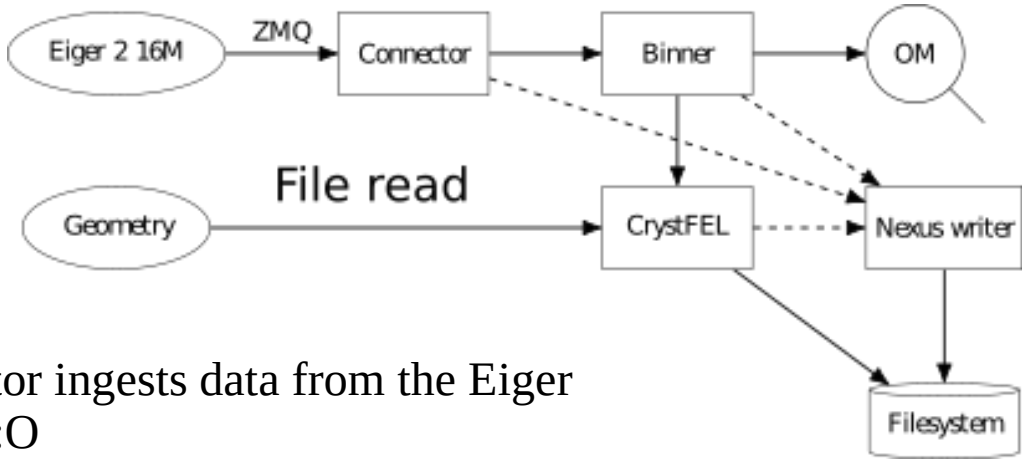






Pipeline step	Delay time
Integration	2 secs
Writer	0.597 secs

Pipeline example



- ASAPO-Eiger-Connector ingests data from the Eiger ZMQ stream to ASAP::O
- (Optional) Binner reduces images resolution to speed up later
- processing steps
- CrystFEL for peak search, indexing, and integration
- OM (OnDA Monitor) for live visualization
- Nexus writer can write raw, binned, or filtered (hits only) images and metadata to disk, depending on which data source it is connected to
- Currently, geometry/analysis results are read/written by CrystFEL from/to disk directly

Summary

- Several data-processing pipelines based on ASAPO are used at different experiment-stations at DESY Petra-III facility.
- Data-processing pipelines build with asapo includes re-usable building blocks provided by support team.
- Reliable performance and support have been demonstrated. Over 100 Hz continuous message rate with multiple MB message size.
- Online data processing improves the performance of the beamline, simplifies data post-processing and enable data reduction.