NSLS-II Data Management Framework

Arman Arkilic
Brookhaven National Lab
NSLS-II

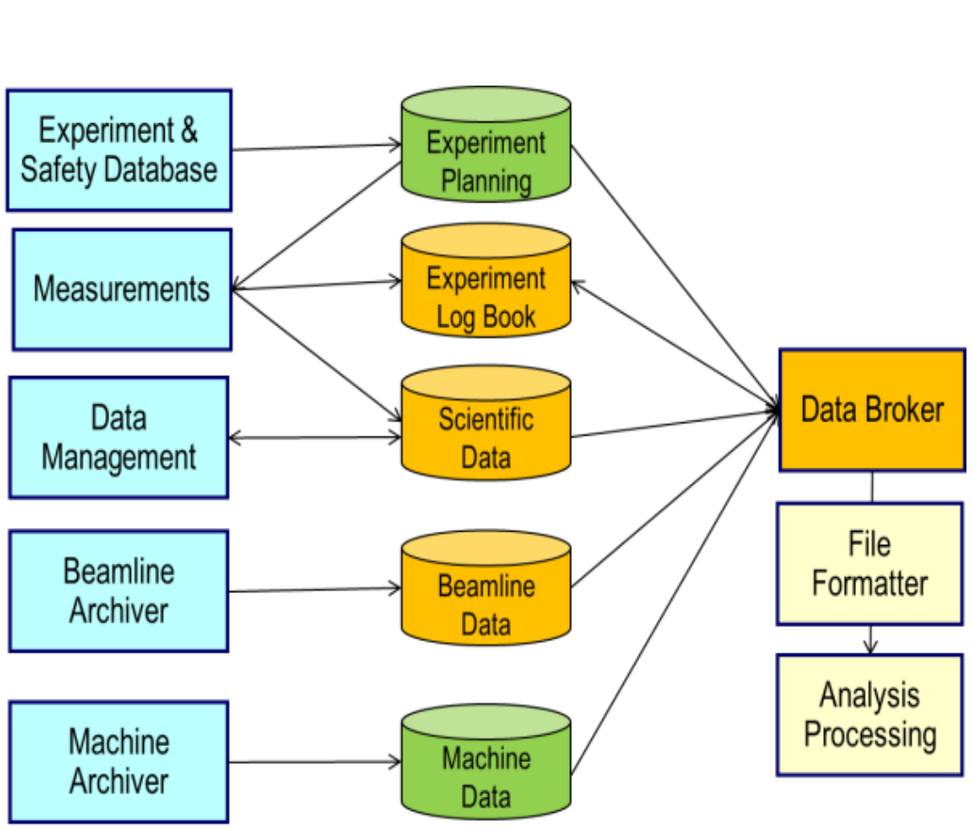
BROOKHAVEN
NATIONAL LABORATORY

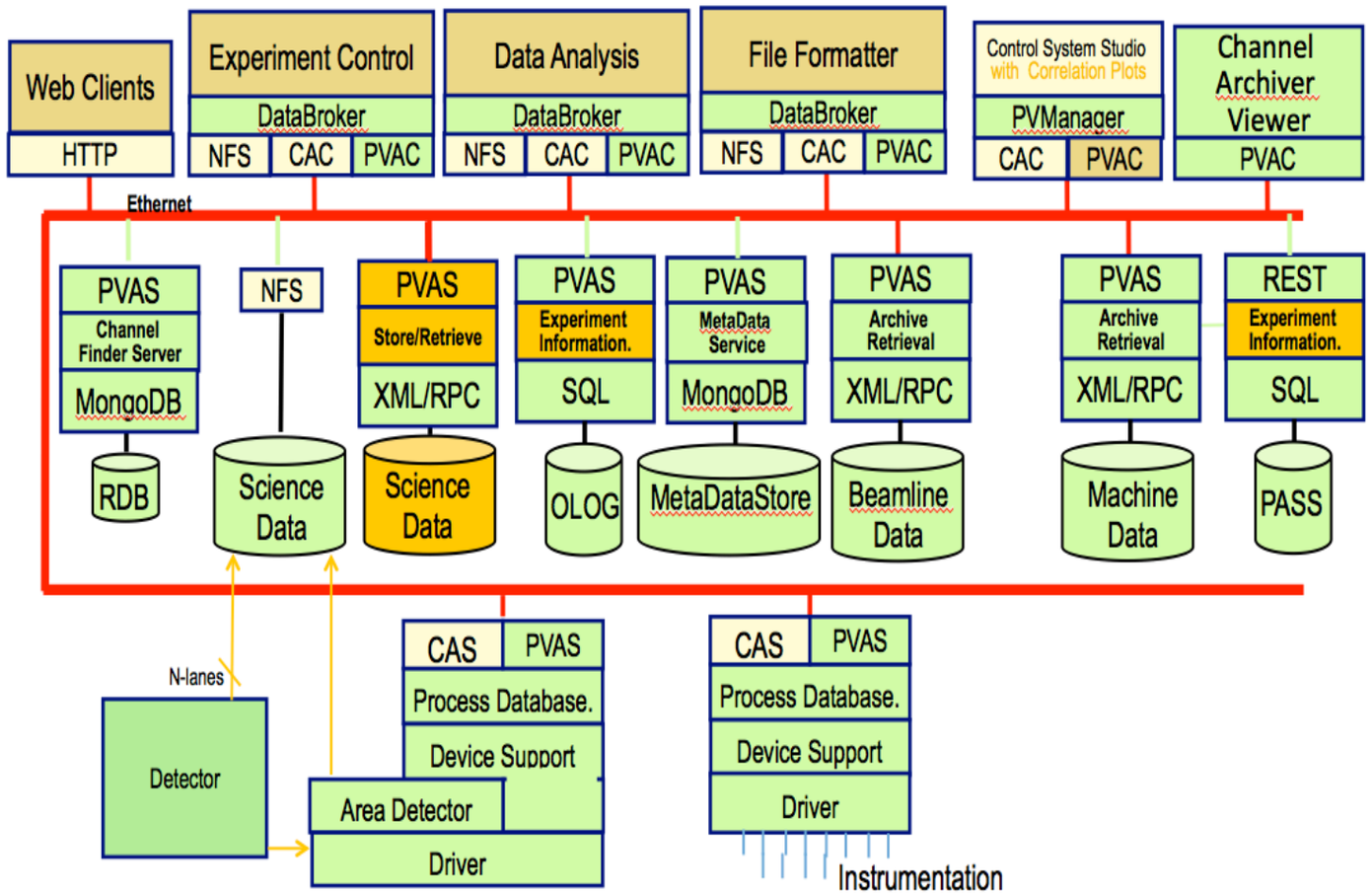
Motivation

- Modern synchrotron experiments require frameworks that provide scientists with data mining/analysis tools
- Given the variety of experiments a good data management framework must accommodate semi-structured and unstructured data
- A centralized framework that serves to multiple scientific applications is more feasible than various frameworks that serve to specific scientific applications for facilities

- File I/O is the bottleneck for most data analysis:
  - Text files are not query-able and impossible to maintain in large scale distributed systems
  - Experiment-specific databases requires reinventing the wheel for each new experiment and can not perform additional tasks that occur with changing needs
  - Text files do not support RESTful interfaces(i.e. one can't easily develop a web service as they would using a database)
  - And pretty much any reason that makes databases better than text files…

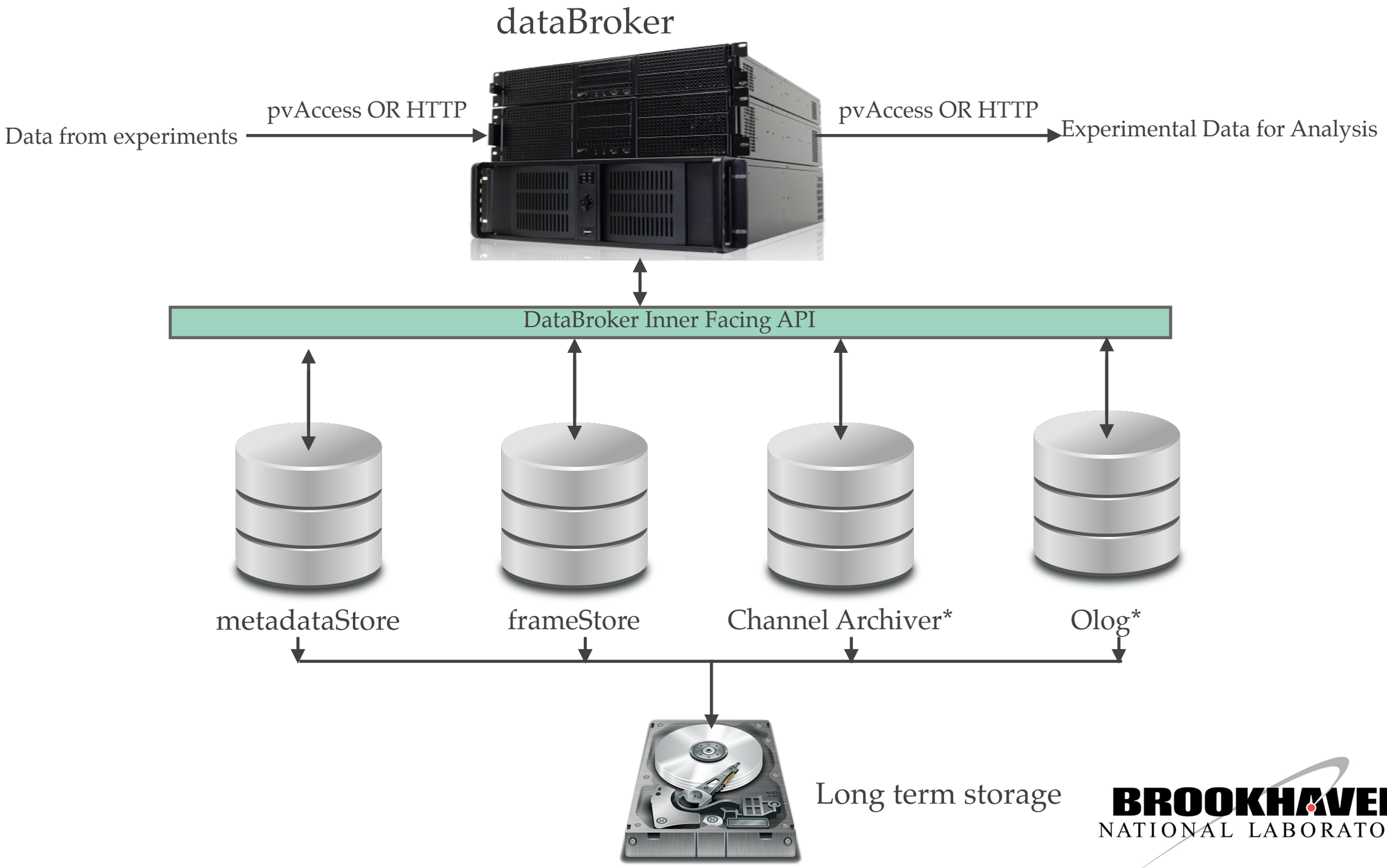
Beamline Service Architecture





BROOKHAVEN NATIONAL LABORATORY

Beamline Data Management Architecture



BROOKHAVEN NATIONAL LABORATORY

Components

dataBroker

dataBroker is a server that provides write and read access to experimental data frameworks underneath it. Acts as a glue for various underlying services

metadataStore

metadataStore is a service that is used in order to record metadata in beamline experiments.

frameStore

frameStore is a service that provides means to manage images and/or any NDimensional data from experiments

Channel Archiver

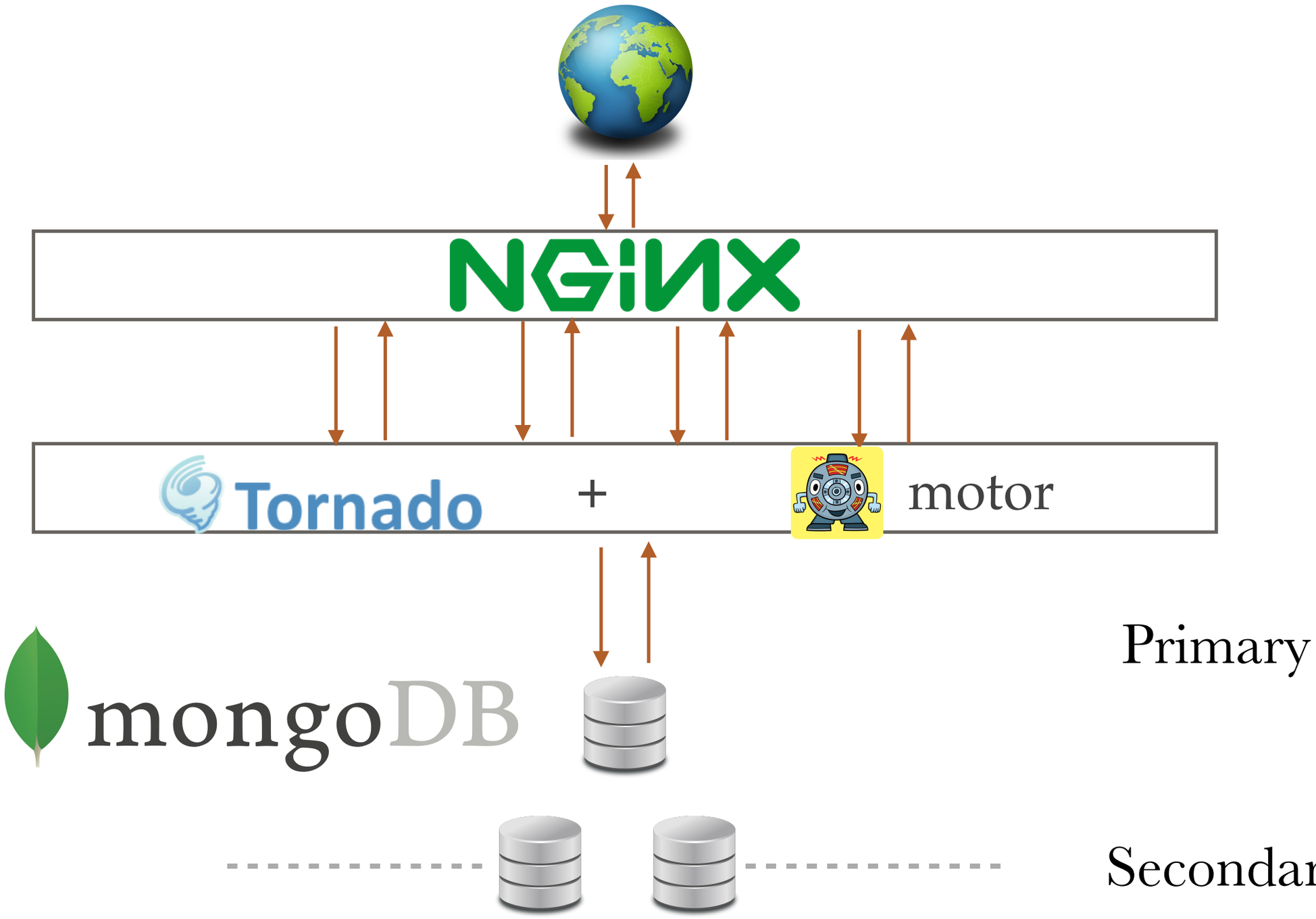
Channel archiver records signals from various hardware and provides means to query time series data

Olog

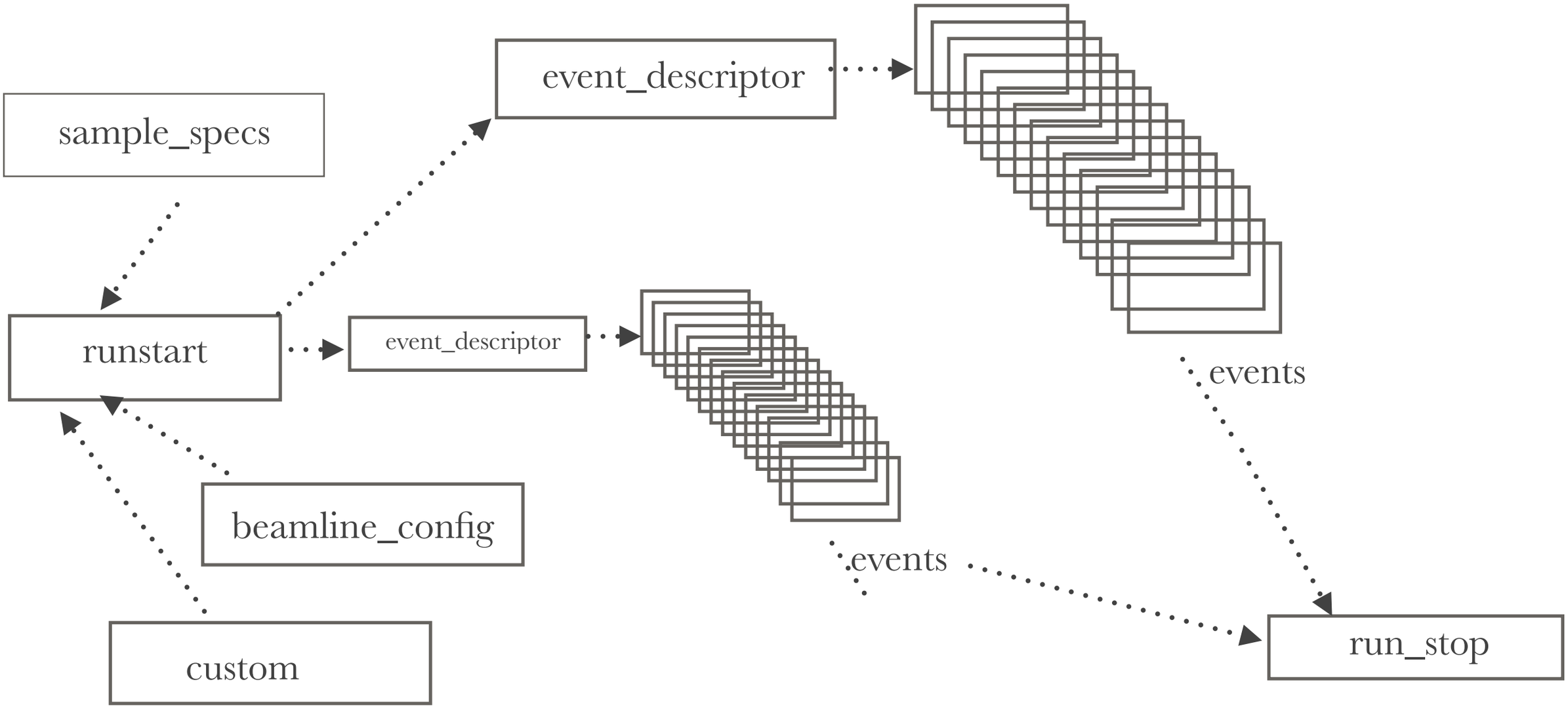
Olog is an operation logbook that keeps track of a user's experimental activity providing various APIs and web clients

BROOKHAVEN NATIONAL LABORATORY

MongoDB Web Service Deployment



metadatastore, not so non-relational…



Middle-layer Services

- The middle layer of the beamline applications are composed of: metadatastore, filestore, olog, archiver appliance, channel finder, analysisstore, and ophyd. databroker and bluesky are the higher level applications that provides users with the ability to control their experiment, while databroker acts as a gateway to experimental data.
- **metadatastore** is the primary source of storage for scans, sweeps, etc. metadatastore is implemented as a RESTful web service on Tornado with a NoSQL mongodb backend. It consists of 4 major collections: RunStart, EventDescriptor, Event, RunStop. One can think of RunStart and RunStop as the head and tail of a series of events that happen throughout an experiment. EventDescriptor(s) contain information regarding the data acquisition Event(s). In other words, they contain information about what are the data_keys that are being captured, what their shapes and data types are.
- This design pattern allows NSLS-II middle-layer to tackle many challenges including asynchronous scans, custom hardware brought from an external source, and varying data formats among experiments.
- **fileStore** is the component of the middle-layer that keeps track of experimental files. These experimental files are generated via EPICS areaDetector module or any custom area detector. file store's NoSQL mongo backend allows users to save any sort of information about their images, while providing links to metadatastore runs.
- This way, any data point in any scan can easily be corresponded to an image, providing scientists with data mining techniques they didn't have access earlier.
- **databroker** is the pathway for data analysis tools to access data without any knowledge of experimental data (implementation of Daron Chabot's architecture). It is implemented in Python and provides users with broker objects that are Pandas data frames, allowing semi and un-structured experiemental data to be accessed within scientific Python framework

databroker

Searching by ID or Recency¶

Here is a summary of the "Do What I Mean" slicing supported by `DataBroker`.

syntax	meaning
`DataBroker[-1]`	most recent header
`DataBroker[-5]`	fifth most recent header
`DataBroker[-5:]`	all of the last five headers
`DataBroker[108]`	header with scan ID 108 (if ambiguous, most recent is found)
`DataBroker[[108, 109, 110]]`	headers with scan IDs 108, 109, 110
`DataBroker['acsf3rf']`	header with unique ID (uid) beginning with `acsf3rf`

Time-based Queries¶

Runs that took place sometime in a given time interval are also supported.

syntax	meaning
`DataBroker(start_time='2015-01')`	all headers from January 2015 or later
`DataBroker(start_time='2015-01-05', end_time='2015-01-010')`	between January 5 and 10

Complex Queries¶

Finally, for advanced queries, the full MongoDB query language is supported. Here are just a few examples:

syntax	meaning
`DataBroker(sample={'$exists': True})`	headers that include a custom metadata field labeled 'color'
`DataBroker(scan_type={'$ne': 'DeltaScan'})`	headers where the type of scan was not a `DeltaScan`

Production Library Performance

- **110K run_start/event_descriptor and 657 million events**
- **275GB of data in total**
- **Finding a unique descriptor or runstart takes ~180 microseconds**
- **Finding events given descriptor(the most common application) takes ~ 43ms**
- **Finding a runstart out of 110.6K takes ~2.5 ms**

For more information:

- http://nsls-ii.github.io/index.html

BROOKHAVEN
NATIONAL LABORATORY

Acknowledgements

- Bob Dalesio
- Daron Chabot
- Daniel Allan
- Thomas Caswell

BROOKHAVEN
NATIONAL LABORATORY