

Streaming Data Securely Anywhere with Tiled Websockets

Daniel Allan

Data Engineering Group Lead

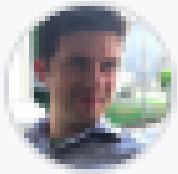
Data Science and Systems Integration Program, NSLS-II

November 2025

Preamble



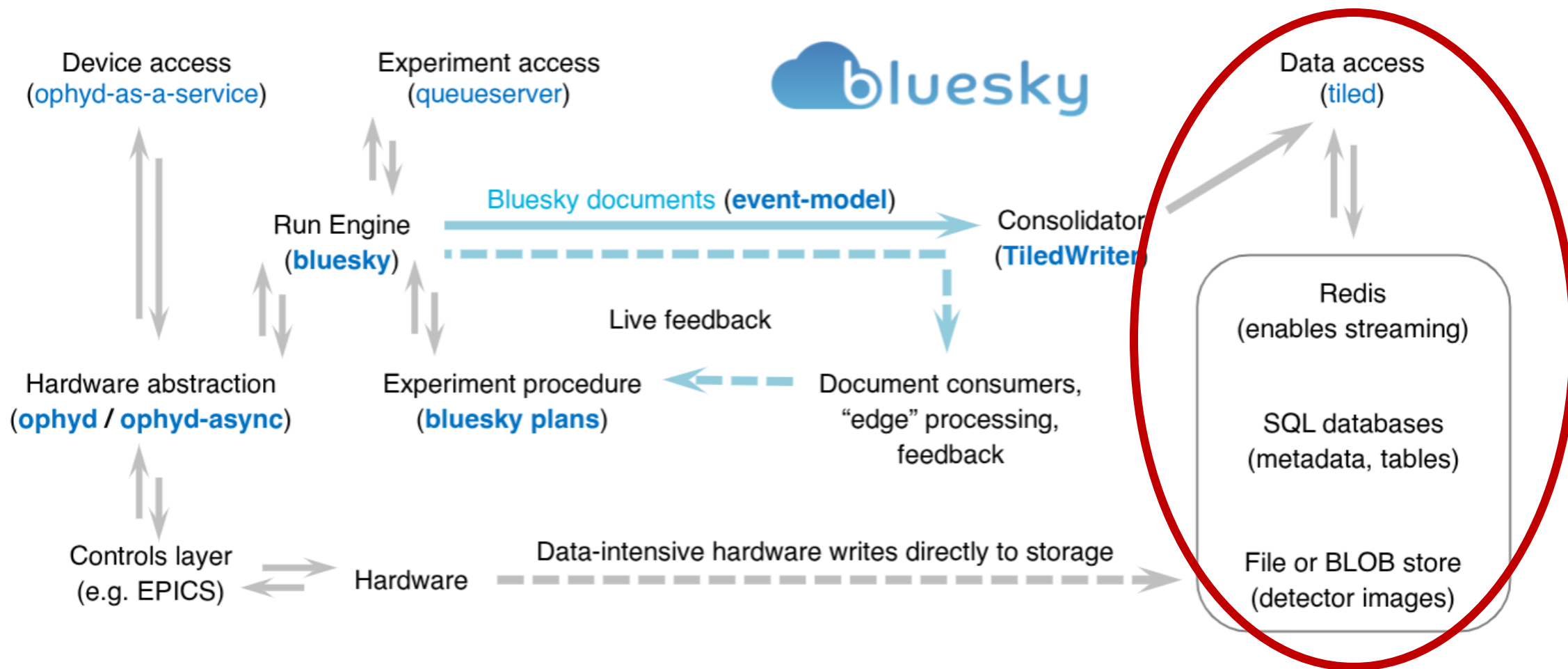
Preamble



danielballan / **Blue Sky Demo.ipynb**

Last active 10 years ago

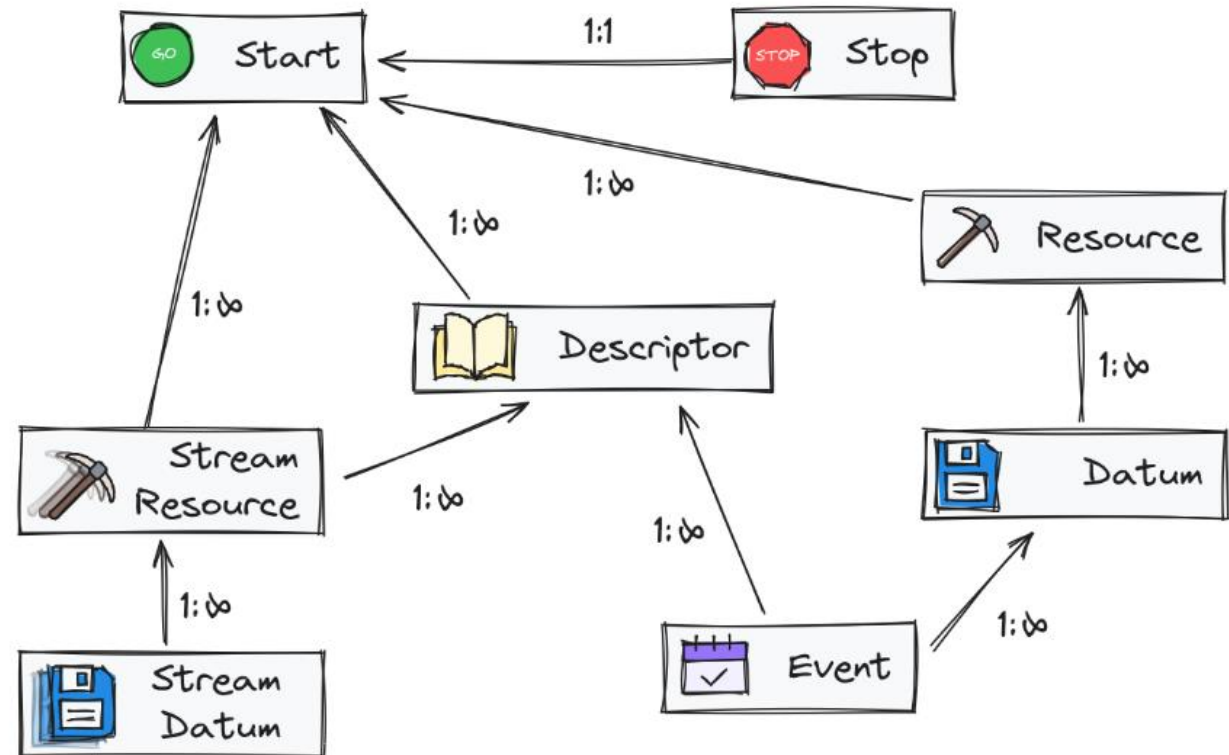
Draft Updated Diagram of Bluesky Core



Building streaming-enabled tools has required understanding Bluesky's "document model"

The model does its job well...

...but it can be a lot to take in...especially for users who do not interact with it often.



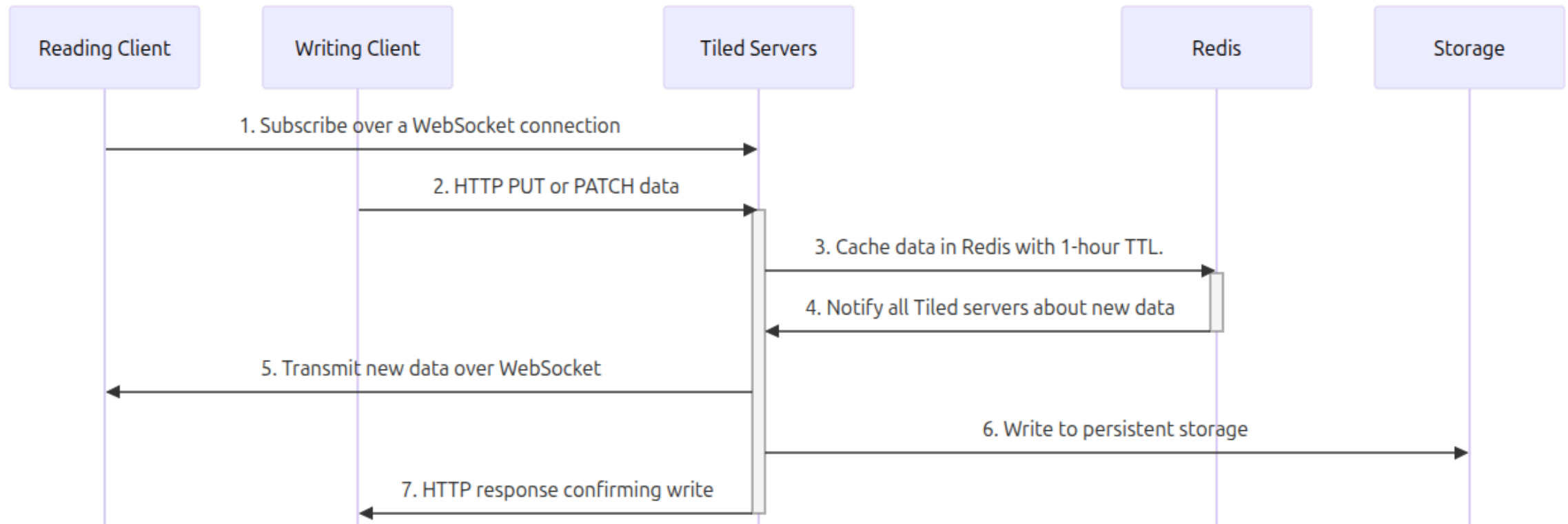
Tiled's Websockets API is simpler than the Bluesky document stream

1. Users see *arrays* and *tables* with metadata.
2. The server takes care of reading the data from detector-written files.
 - Keep data analysis software environments lightweight
 - Makes data accessible in environments where the formats are not supported (like a web browser) or filesystems are not reachable
3. Secure by default, safe and easy to expose externally
 - Unlike raw Redis, Kafka, EPICS CA or PVA
 - Let intermediate HTTP proxies provide load balancing, rate limits, ...
 - We work hand in hand with our local Cybersecurity experts on this (and you should too!)

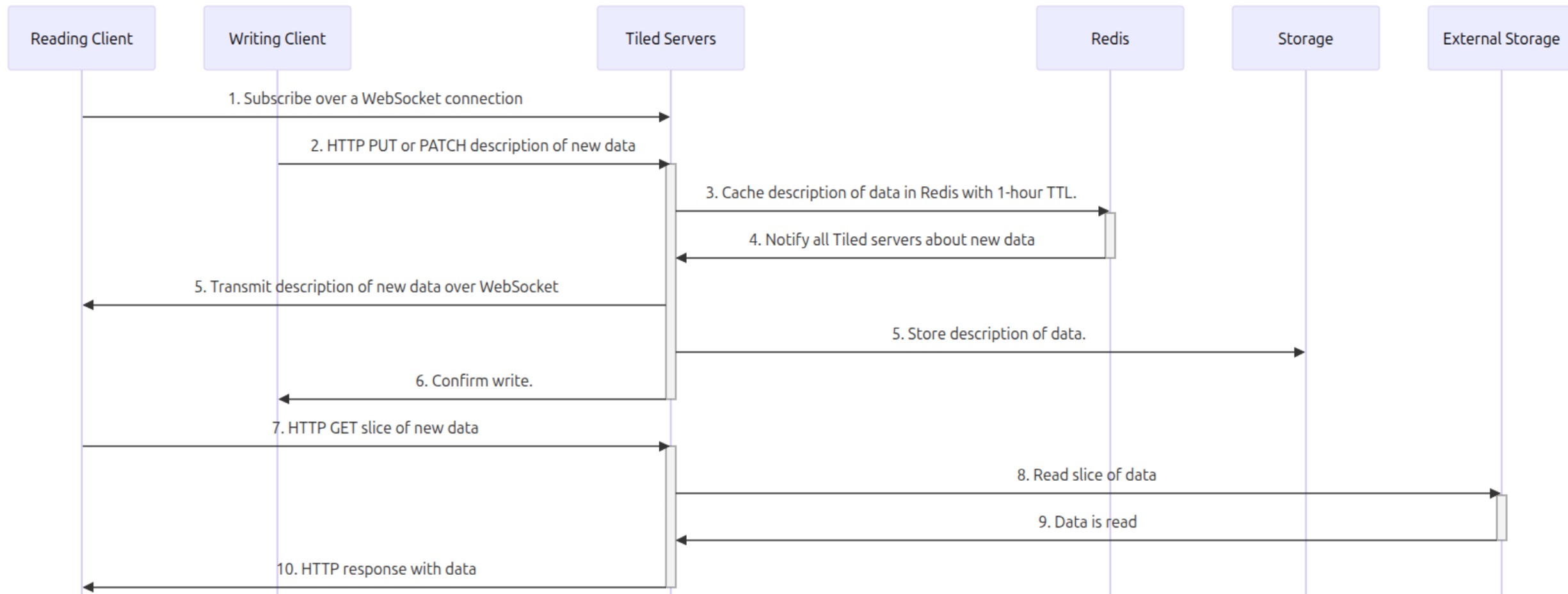
Tiled Websockets API complements the Bluesky document model

- The RunEngine will (of course) continue to emit documents.
- Tools that consume Bluesky documents (e.g. "suitcase") will continue to work.
- But for most users, Tiled Websockets will be more convenient and easier to get started with.
- Going forward, build on Bluesky documents only if you need tight in-the-loop integrations that cannot tolerate ~100ms of latency.

How Tiled Websockets Broadcast Uploaded Data



How Tiled Websockets Broadcast Registered Data



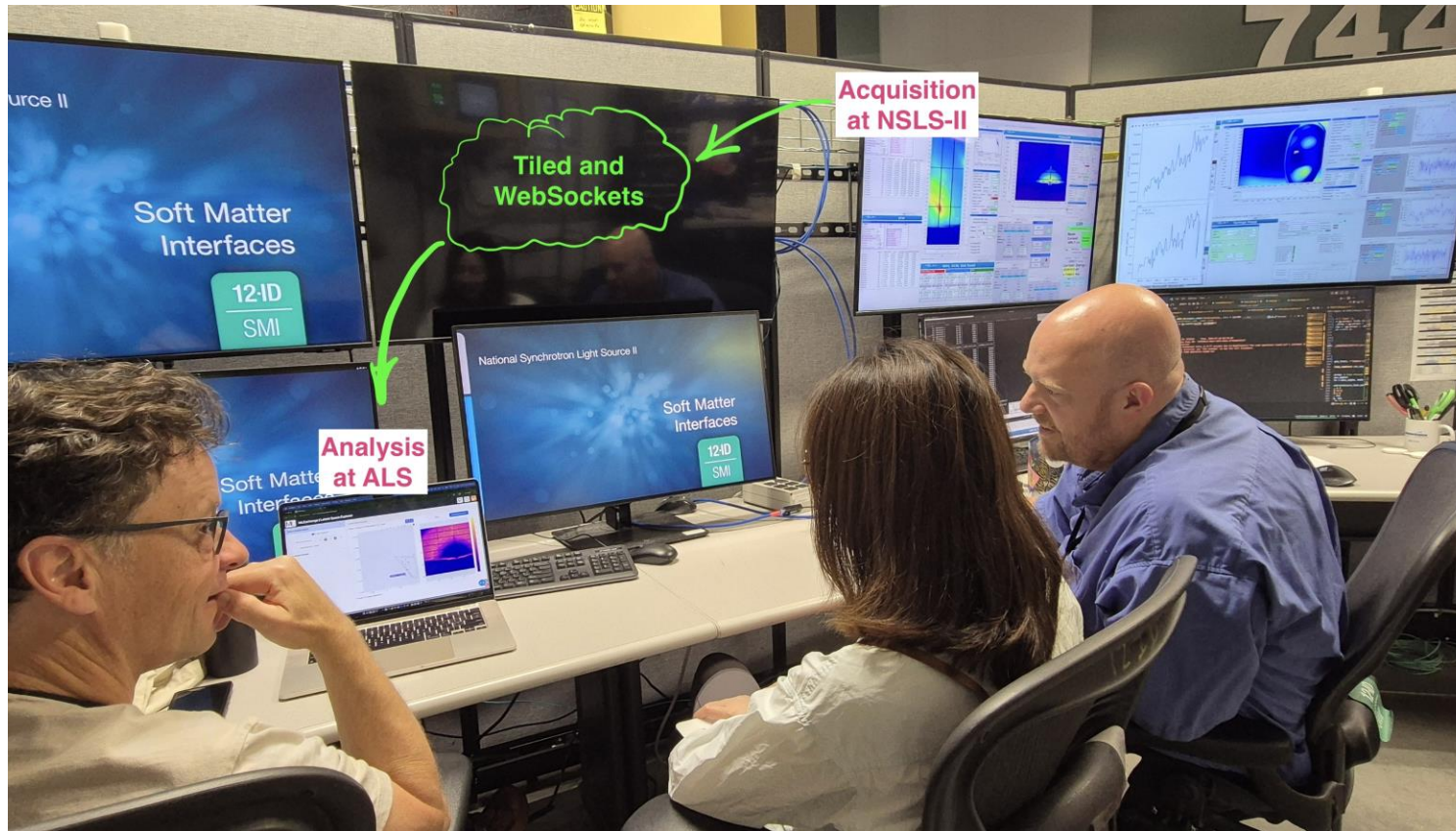
Tiled Websockets API Highlights

- For now, one WS connection per dataset. A **multiplexing** endpoint is planned.
- Server can send **JSON** (universal) or **msgpack** (more efficient).
- Server can **replay past updates** that the client missed, up to what is still cached.
- After connecting, an initial **"schema" message** includes:
 - A **protocol version number**, so we can change things later
 - Context that is static over the connection lifecycle (e.g. array data type or table schema)
- Each **update message** includes:
 - **Timestamp** of when the data was cached, to communicate latency
 - **Sequence number**, to facilitate resuming after an interruption
 - **Format** (mimetype) of data payload, if applicable

For a user in Python this looks like...

```
>>> from tiled.client import from_uri
>>> client = from_uri("https://tiled.nsls2.bnl.gov")
>>> sub = client["path/to/my/dataset"].subscribe()
>>> sub.new_data.add_callback(print)
>>> sub.start()
```

We live-streamed data from NSLS-II to ALS to support a real user experiment

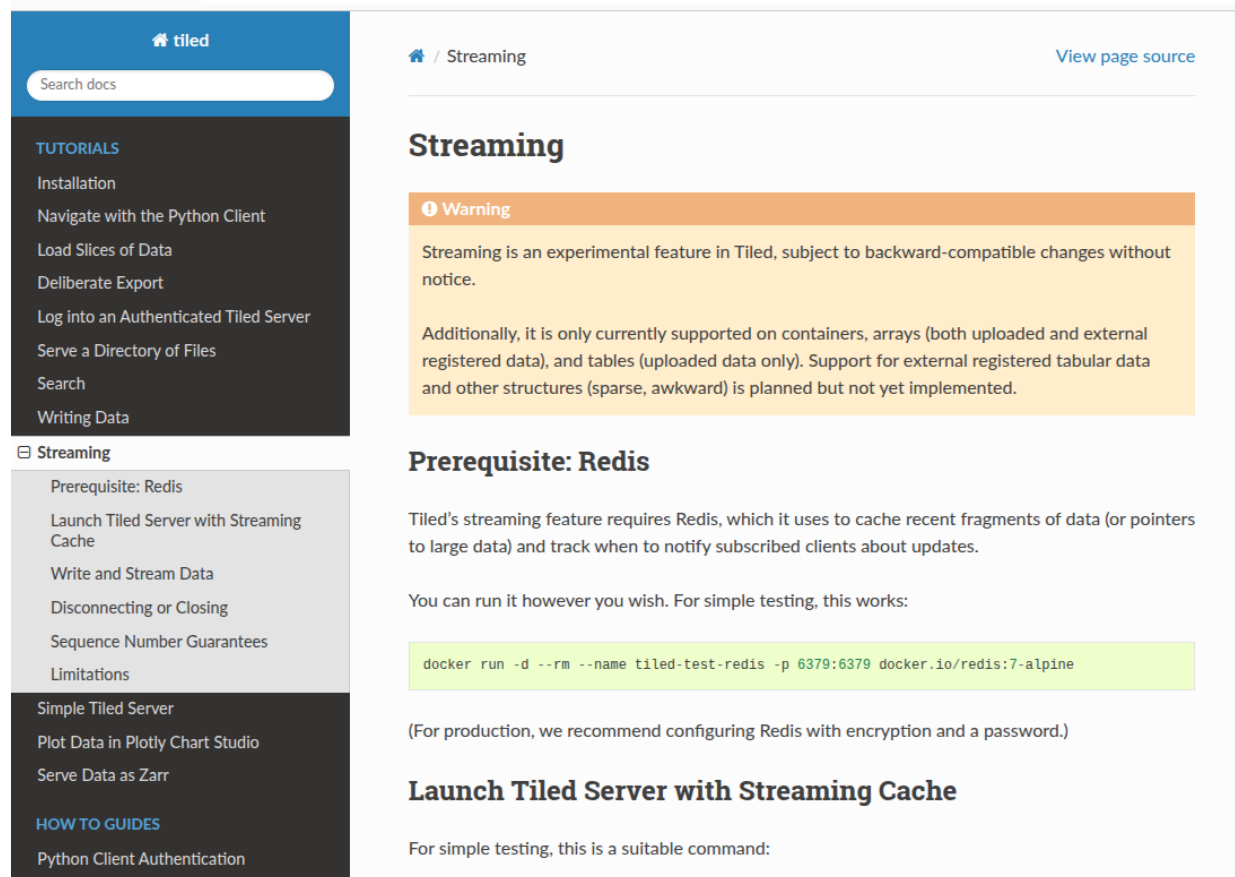


- Streaming data off site has required...
 - Software tailored to a specific detector
 - Special security accommodations (exceptions)
- *Tiled* can now stream scientific data over an industry-standard protocol (WebSockets) that flows securely through firewalls and proxies.
- Enables science by removing organizational friction and coordination overhead

Next—with anyone who wants to help!

- Integrate the streaming support into our plugins for the PyMCA and Napari desktop applications
- Integrate streaming support in web-based applications, starting with bluesky/finch
- Reimagine LiveTable, LivePlot, LiveFit, and Best-Effort Callback as consumers of arrays and tables over Websockets
 - We will not change the existing (document-based) tools that have been stable for a decade.
 - We aim to soon *recommend* new tools, yet to be built.

Documentation is at blueskyproject.io/tiled



The screenshot shows the Tiled documentation website. On the left is a dark sidebar with a search bar and navigation links. The main content area is titled 'Streaming' and includes a warning box, a prerequisite section for Redis, and a code block for launching the server.

tilde

Search docs

TUTORIALS

- Installation
- Navigate with the Python Client
- Load Slices of Data
- Deliberate Export
- Log into an Authenticated Tiled Server
- Serve a Directory of Files
- Search
- Writing Data

Streaming

- Prerequisite: Redis
- Launch Tiled Server with Streaming Cache
- Write and Stream Data
- Disconnecting or Closing
- Sequence Number Guarantees
- Limitations

Simple Tiled Server

- Plot Data in Plotly Chart Studio
- Serve Data as Zarr

HOW TO GUIDES

- Python Client Authentication

Streaming

[View page source](#)

Warning

Streaming is an experimental feature in Tiled, subject to backward-compatible changes without notice.

Additionally, it is only currently supported on containers, arrays (both uploaded and external registered data), and tables (uploaded data only). Support for external registered tabular data and other structures (sparse, awkward) is planned but not yet implemented.

Prerequisite: Redis

Tiled's streaming feature requires Redis, which it uses to cache recent fragments of data (or pointers to large data) and track when to notify subscribed clients about updates.

You can run it however you wish. For simple testing, this works:

```
docker run -d --rm --name tiled-test-redis -p 6379:6379 docker.io/redis:7-alpine
```

(For production, we recommend configuring Redis with encryption and a password.)

Launch Tiled Server with Streaming Cache

For simple testing, this is a suitable command: