

National Synchrotron Light Source II



Bluesky: A Collaborative Toolbox

Daniel Allan

Data Science and Systems Integration Program, NSLS-II

Middle Layer Brainstorming Session, February 2024

Part 1: *Bluesky* Approach

What is Bluesky?

- An **open-source** Python-based toolbox for scientific data acquisition
- A bridge between the Scientific Python ecosystem and facility-scale or bench-scale experimental data acquisition setups
- Mini "ecosystem" of modular Python libraries
 - Python **hardware abstraction** above EPICS (or Tango, or custom)
 - Experiment **sequencing engine**
 - Portable **experiment logic** ("plans")
 - Streaming-first, metadata-rich **data model, unopinionated about data formats**
 - A growing number of libraries, services, and applications built on these

What do we mean by "open source"?

1. Publicly visible source code
2. Licensed for reuse with an OSI-approved license
3. Accepting contributions
4. Open development
5. Open decision making
6. Multi-institution engagement



from Matt Rocklin's post
<https://www.coiled.io/blog/stages-of-openness>

Who uses Bluesky?

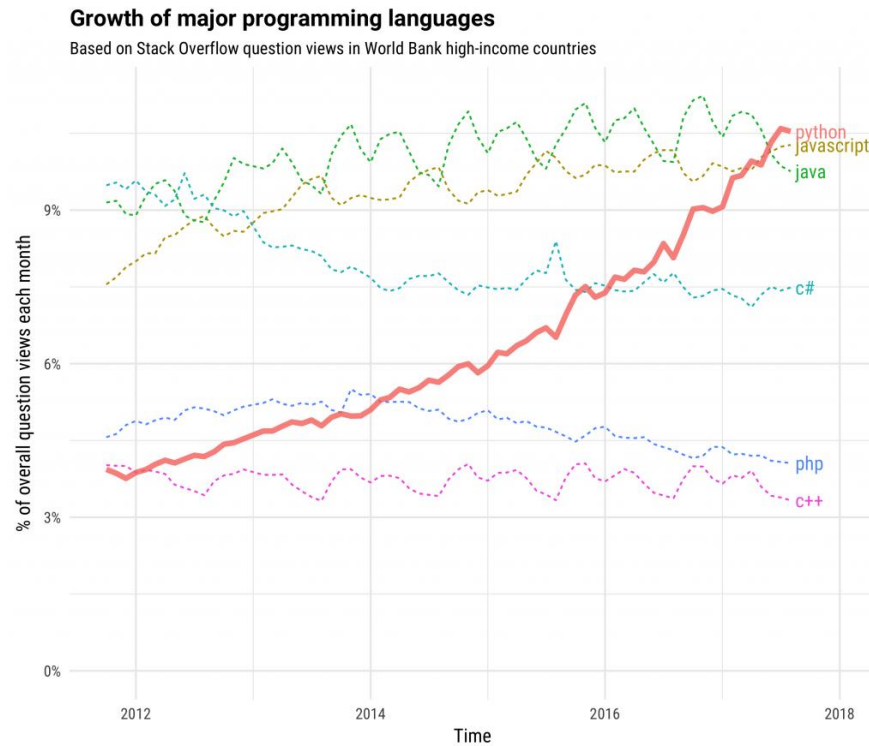
- Significant adoption at...
 - 5 DOE light sources
 - Australian Synchrotron
 - BESSY-II
 - Pohang Light Source
- Diamond Light Source
 - Multiple FTEs making major contributors to Bluesky for over 2 years
 - Bluesky will underpin all scanning for the new beamlines delivered by Diamond-II upgrade
 - Scope for adoption at existing beamlines is under discussion
- Some adoption at Canadian Light Source
- Being evaluated at CHESS, PSI (Swiss Light Source)
- Various academic labs and other facilities



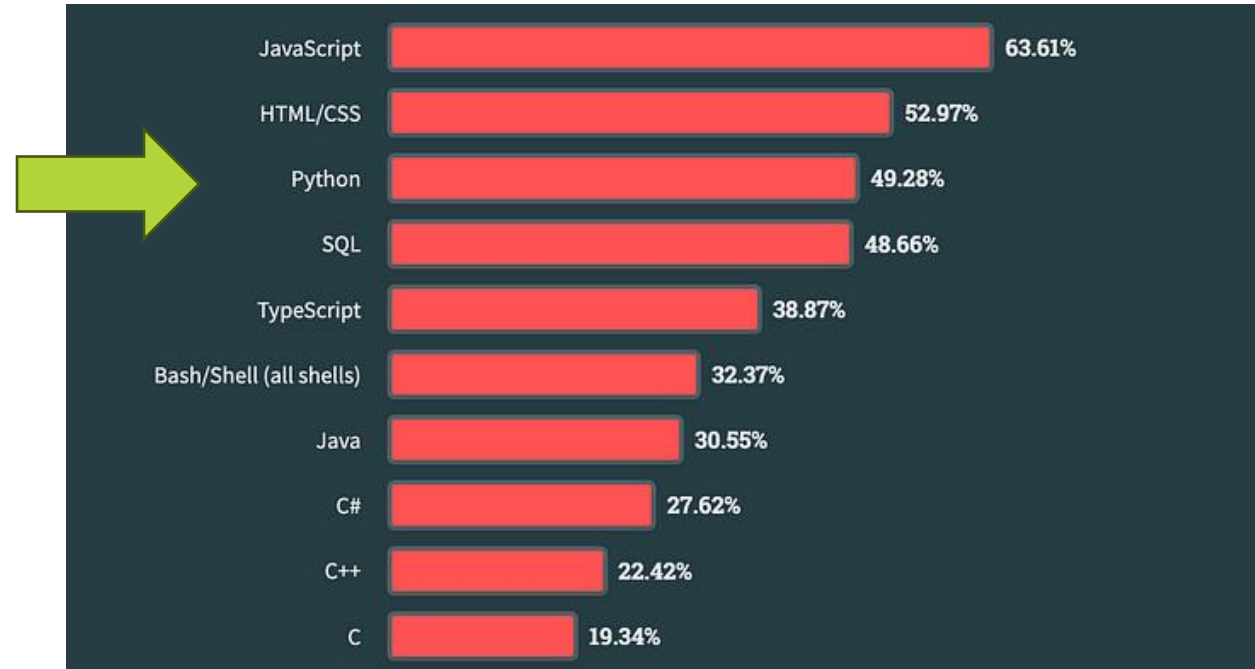
How does collaboration work at scale?

- Adoption can be incremental: take some pieces, leave or customize others.
- This is not an all-or-nothing framework that you have to buy into; it's a **mini-ecosystem of co-developed but individually useful tools** that you can build on.
- It's all in **Python**. Some beamline staff and partner users have built on it.
- The scientific Python community is an example of how distributed, loose collaboration can work well.

Why Python? It is a widely-known language with a growing, friendly community



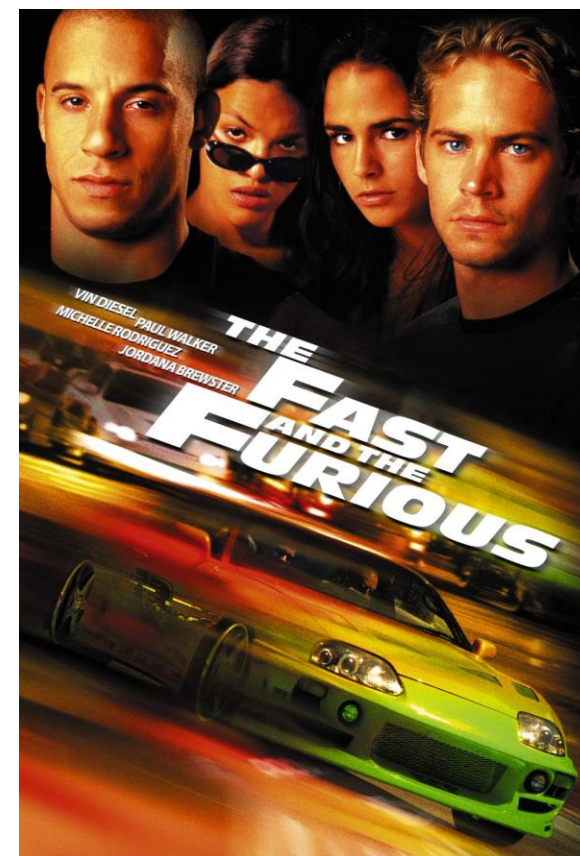
2012-2018: Growth of Python for Data Science



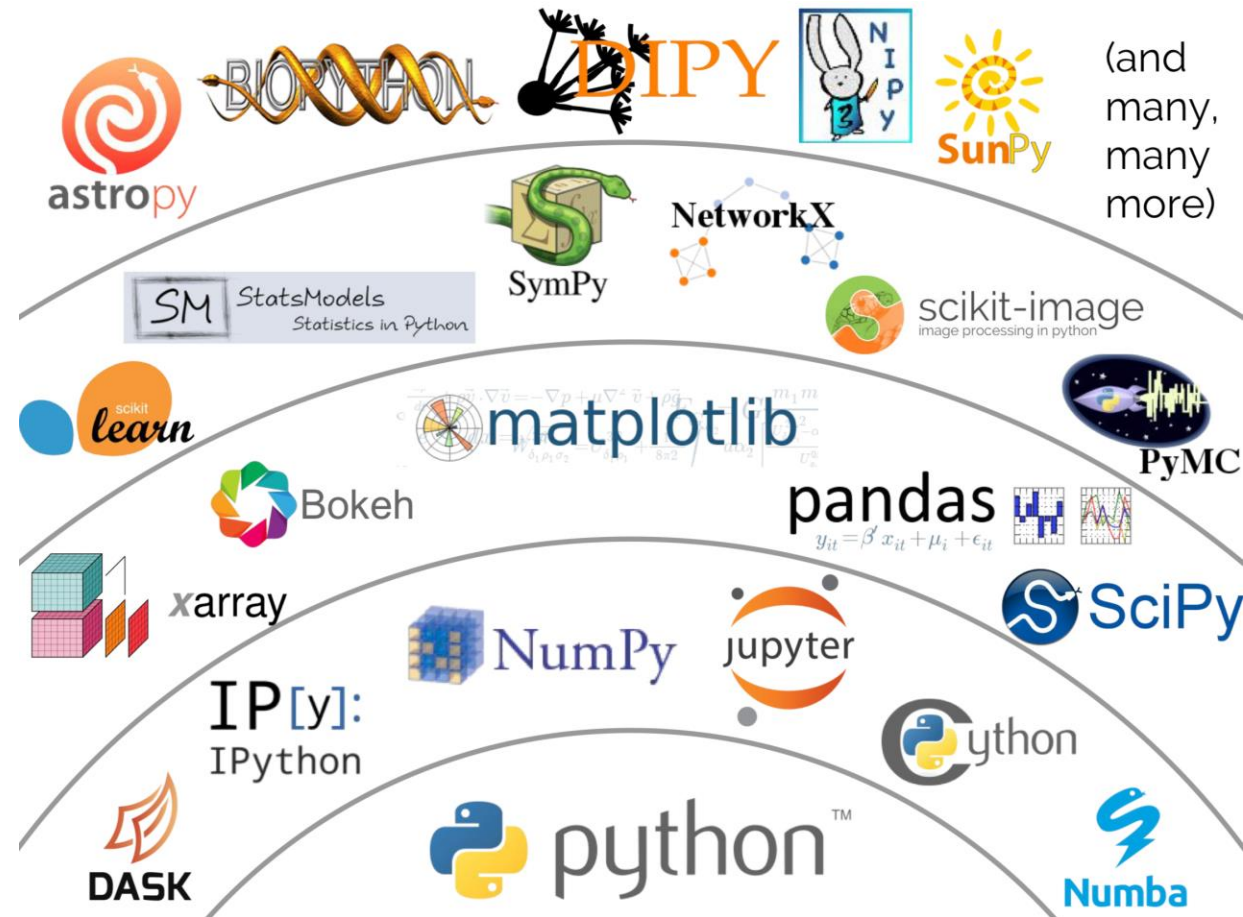
2023: Python "takes center stage" as a friendly and versatile programming language, now often taught in introductory scientific computing courses

Python is not the bottleneck for speed

- Python is just the "glue".
- Expensive numerical computations happen in C or Fortran, wrapped by Python.
- Very often, the actual bottlenecks are waiting for hardware to move, or for data to transfer from disk or over the network.



Scientific Python is a *layered* ecosystem



Some Technical Goals of Bluesky (1 / 2)

- Working backward the needs of data analysis, capture all the important context.
- Be unopinionated data *formats*; focus on data *structures*.
- Support streaming.
- Handle asynchronous data streams.
- Support multi-modal: simultaneous, cross-instrument, cross-facility experiments.

Some Technical Goals of Bluesky (2 / 2)

- Form a bridge to the successful scientific Python ecosystem.
- Be generic across science domains.
- Scale up to facility-scale.
- Scale down to lab bench deployments.
- Be easy for a science grad student to install and try.

Some Sociological Goals of Bluesky

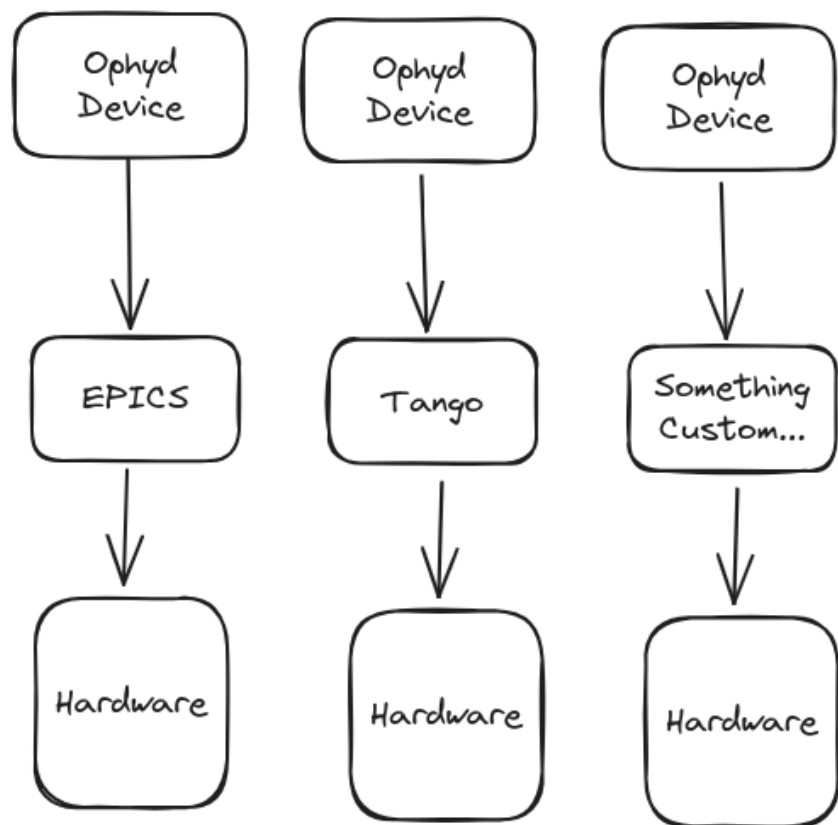
- Overcome "not-invented-here"-ism.
- Make components with well-defined boundaries, *co-developed but individually useful*, which can be adopted piecemeal by groups and facilities.
- Enable code to be reused in ways unforeseen by the original authors.
- Enable beamlines to present their users with a fine-tuned, polished user experience.

Share the toolbox



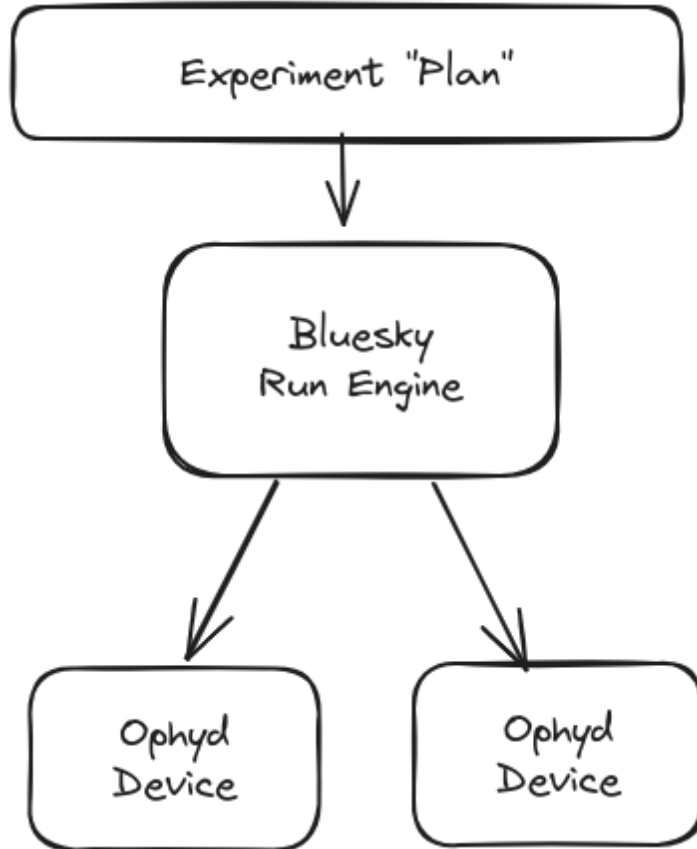
Part 2: *Bluesky* Architecture

Ophyd is a Python hardware abstraction



Provide high-level verbs like "trigger", "read", "set", "stop", ... over top of lower-level commands in EPICS or other control systems

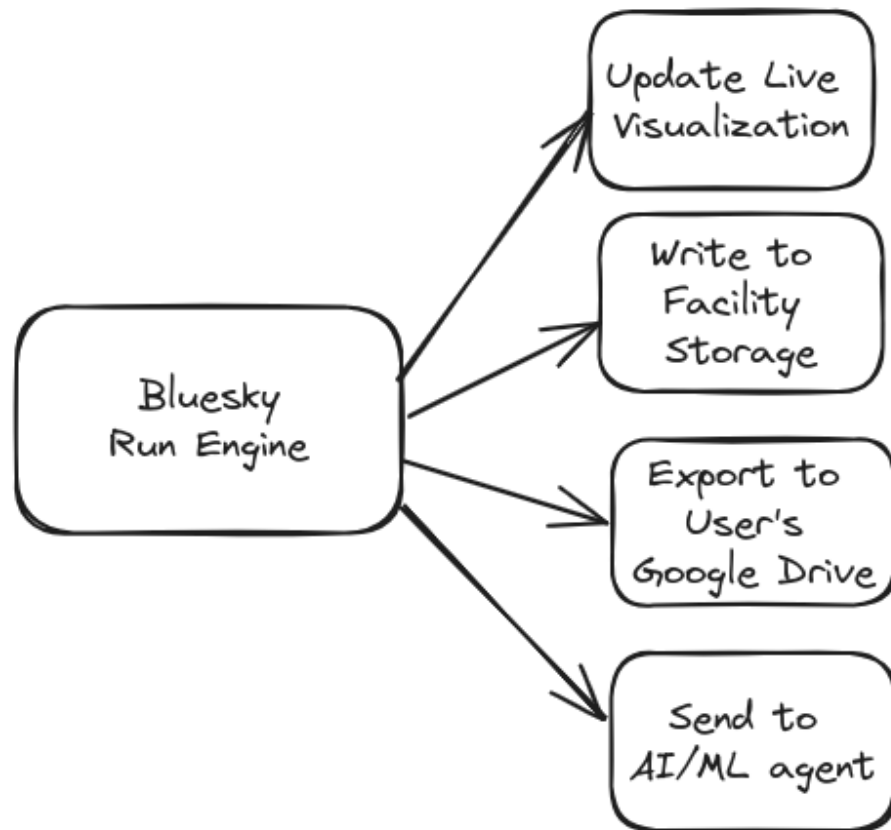
Bluesky *Run Engine* orchestrates experiments



Execute portable "plans" on given hardware, handling feedback (e.g. progress bars), interruptions, and failure modes.

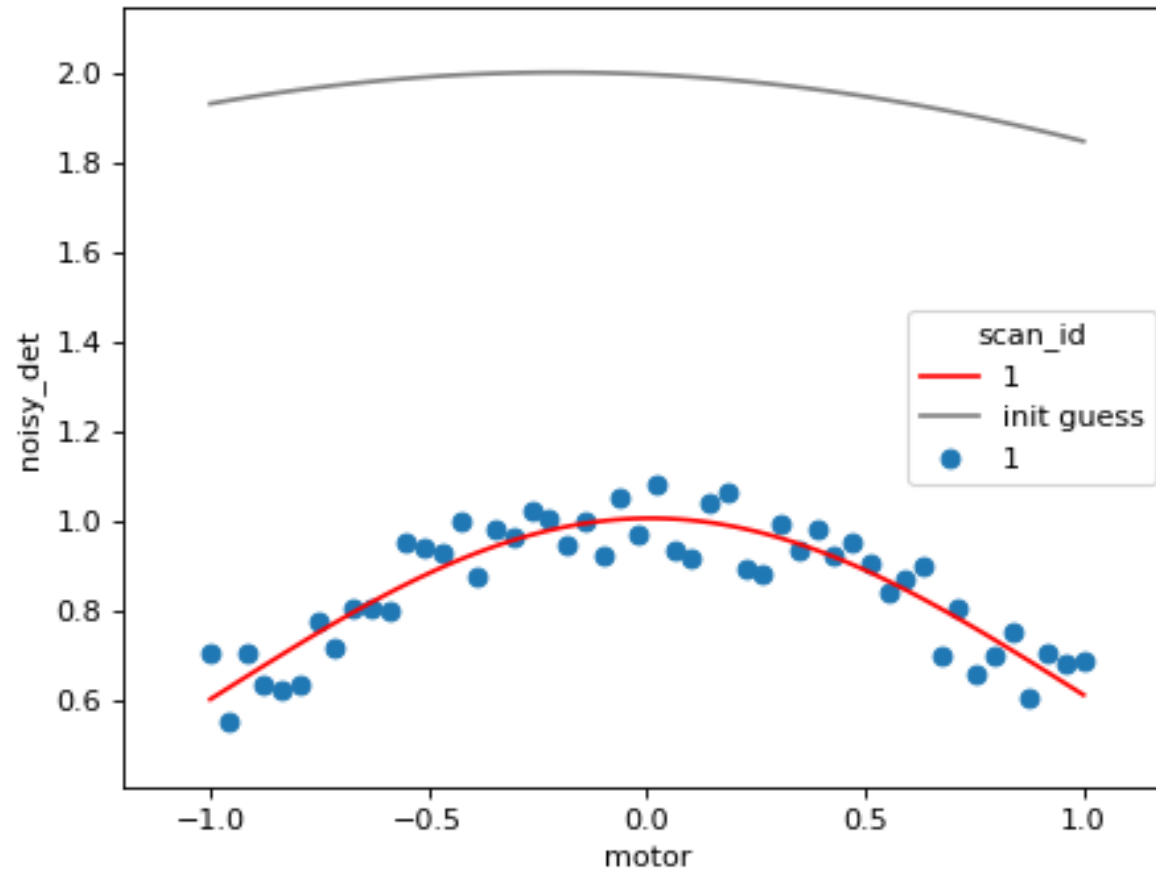
Simulation is typically plugged in via Devices that implement simulations instead of actual hardware I/O.

Bluesky *Run Engine* broadcasts metadata and data to consumers



Stream metadata and data to a configurable set of consumers.

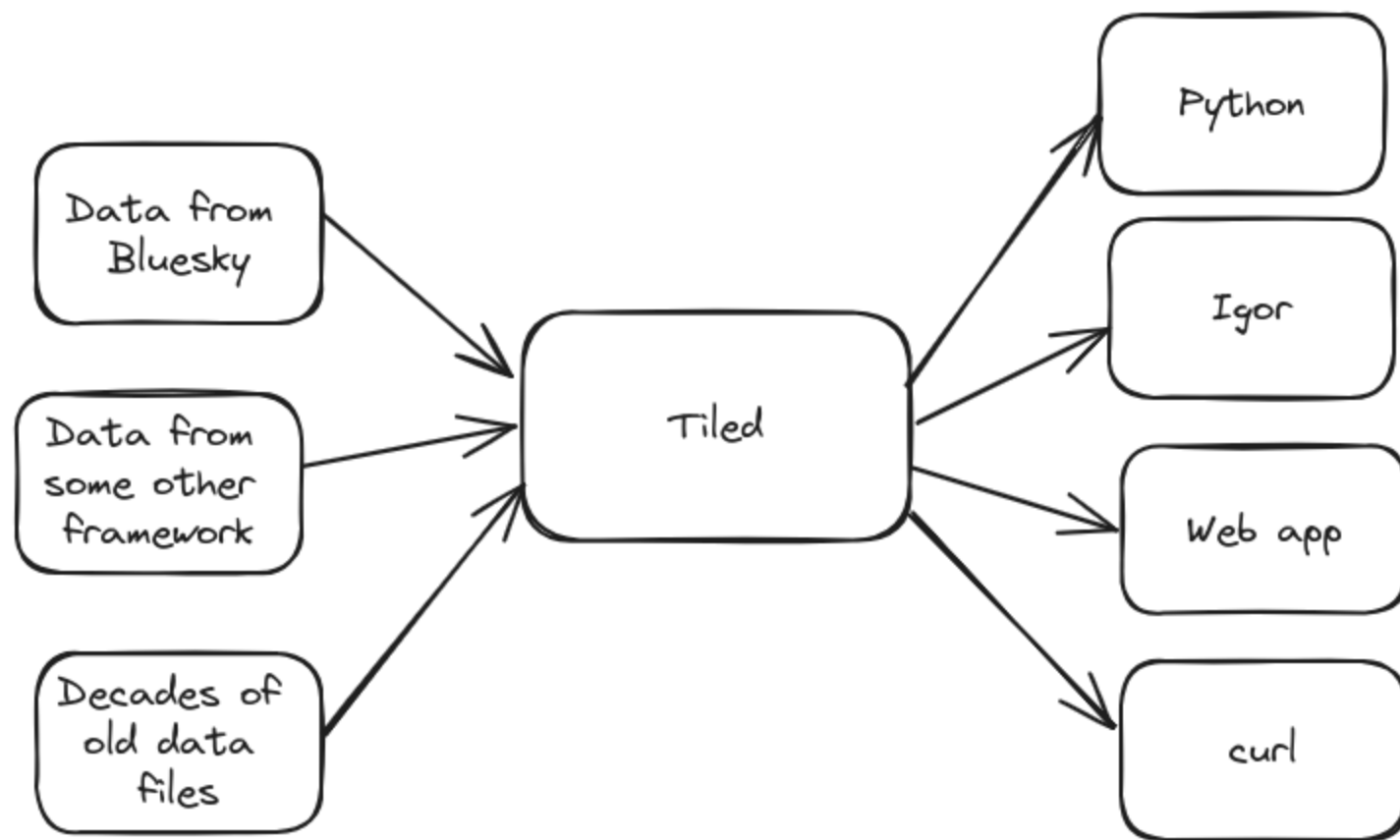
Example: Live Fitting and Visualization



Data and Metadata are emitted in a streaming fashion in Documents

- Documents are JSON, a machine- and human-readable standard
- Follows a lightweight schema with open-ended space for scientific metadata
- Small data can go directly in the document; large data is referenced (e.g. a filepath and location in file)
- How data is *chunked* into these documents matters.
 - To make things go fast, careful choices need to be made here.
 - New work, in collaboration with Diamond Light Source, is focused on high-performance (fast) configurations for fly-scanning.

Tiled is the future of Databroker



Make data searchable, sliceable, and accessible in many formats, regardless of how it is stored

Tiled generalizes Databroker:

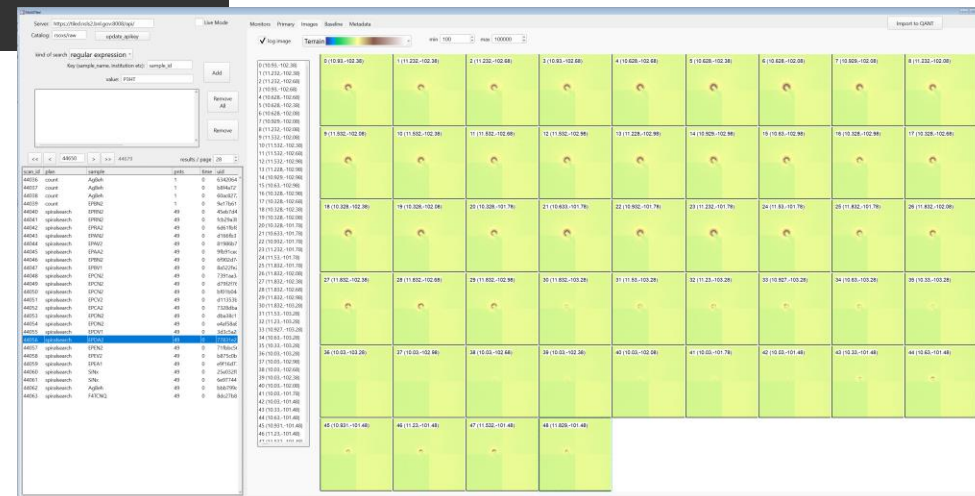
- Not just Python
- Not just data that came from Bluesky

Data directly into an analysis program

```
In [1]: from tiled.client import from_uri
In [2]: c = from_uri("https://tiled-demo.blueskyproject.io/api")
In [3]: c
Out[3]: <Node {'generated', 'csx', 'bmm', 'fxi', 'um2022'}>
In [4]: run = c['bmm']['raw'].values().last()
In [5]: run
Out[5]: <BlueskyRun {'baseline', 'primary'} scan_id=32450 uid='10cefec0' 2022-03-27 09:29>
In [6]: run['primary']['data']['IO'][0][:5]
Out[6]: array([68.24125113, 68.09572678, 68.09231882, 67.90571976, 67.71874719])
```

A custom (scientist-written!) Igor program loads data from Tiled over HTTP, integrating search and viz

Tiled's officially-supported Python client loads data efficiently into numpy, pandas, xarray, awkward



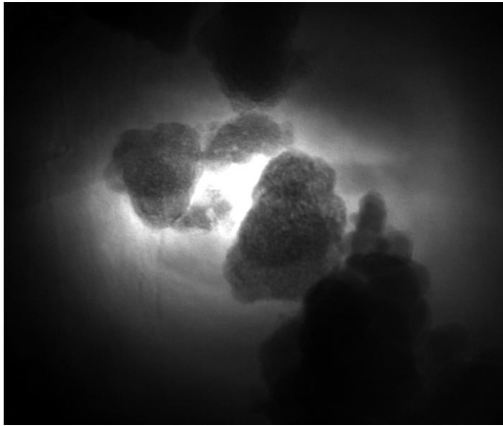
Data in a web browser, on a phone

TILED BROWSE

Top / fxi / raw / 1b0b4d73-6d87-43ab-8d62-ed035c51b9b4 / primary / data / Andor_image

VIEW DOWNLOAD METADATA DETAIL

Andor_image



Choose a planar cut through this 4-dimensional array.

This large array has been downsampled by a factor of 2. Use the "Download" tab to access a full-resolution image.

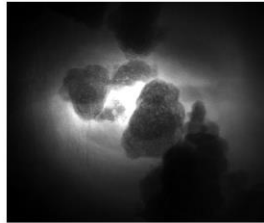
0 22 44
0 10 19

TILED BROWSE

Top / fxi / raw / 1b0b4d73-6d87-43ab-8d62-ed035c51b9b4 / primary / data / Andor_image

VIEW DOWNLOAD METADATA DETAIL

Andor_image



Choose a planar cut through this 4-dimensional array.

This large array has been downsampled by a factor of 3. Use the "Download" tab to access a full-resolution image.

0 22 44
0 10 19

TILED BROWSE

Top / fxi / raw / 1b0b4d73-6d87-43ab-8d62-ed035c51b9b4 / primary / data / Andor_image

VIEW **DOWNLOAD** METADATA DETAIL

Dimensions: 45 × 20 × 1080 × 1280


Slice (Optional) [EXAMPLES](#)

If blank, access entire array

Format *
CSV

DOWNLOAD **LINK** **OPEN**

Link
<https://tiled-demo.blueskyproject.io/a>



Tiled can provide slices of data as...

- Custom, one-off text format designed to be parsed by a **30-year-old bash script** that still works ("if it ain't broke...")
- Traditional formats like CSV, TIFF, or HDF5 to be opened by **Igor, Origin, ImageJ, PyMCA,**
- **Web-friendly** image or data formats to be directly displayed by a web browser or web application
- Chunks of compressed **C** or **Arrow**-encoded buffers to be fetched on demand and fed zero-copy to **Tensorflow**



We can meet all users where they are!

Two Modes of Use for *Tiled*

- Tiled as active source of truth
 - Write metadata and data into Tiled (in whatever format) over HTTP
 - Tiled stores it in an opinionated format
 - SQL database + files comprise the source of truth, "managed" by Tiled
- Tiled as passive cache, alongside files
 - Read-only view of existing data
 - Take formats as they are
 - Index file metadata, structure, location in database
 - Coexists peacefully with existing file-based workflow



Wrapping up...

- *Bluesky* is a multi-facility open-source collaboration.
- It is a toolbox, not a ready-to-use application.
- The core Python libraries are well established and have been stable since 2017.
- blueskyproject.io



Questions & Discussion

Extra Slides...

Areas of Active Development (1 / 1)

- **Graphical Applications**
 - Integrations with applications our users already use
 - New applications
- **Services**
 - Run Engine as a service (Queueserver, Blue API)
 - Data Access (Tiled)



Areas of Active Development (2 / 2)

- Lean into **Autonomous Experiments** and Adaptive Control Logic
 - Bluesky has always supported this, but new developments exploit this more and make it easier
- Enrich **Fly-scanning**
 - Currently, Bluesky just gets out of the way ("B.Y.O. fly-scan")
 - Led by Diamond, the collaboration is building fly-scanning components that do more right out of the box

Minimum Viable Governance

- Maintainers: per repo, make day-to-day decisions and set processes as appropriate to the repo
- Technical Steering Committee: arbitrate when maintainers cannot reach rough consensus
- Project Advisory Board: management-level stakeholders, oversee big-picture priorities

