Helmholtz Zentrum Berlin

BLUESKY TRAINING

EPICS COLLABORATION MEETING POHANG 2024

12.04.2024

Marcel Bajdel, Luca Porzio

Helmholtz-Zentrum Berlin, Beamline Optics Department

- HZB is a „User Facility”
- User mail samples or visit to perform experiments
- More than 40 beamlines
- EPICS in the control layer, and Bluesky for the experiment specification & orchestration
- Upgrades on the horizon BESSY II+ and BESSY III

Reach us at:

marcel.bajdel@helmholtz.berlin.de

luca.porzio@helmholtz.berlin.de

Content overview

Meant for beginners with python knowledge. We will focus on the Bluesky as a toolkit and not on the code nuances.

- Introduction to Bluesky
- Ophyd abstraction layer
- 10 min break
- What is Run Engine?
- How to run plans and create custom plans
- Managing suspensions and interruptions
- 10 min break
- Bluesky documents and metadata
- Data processing using callbacks
- Examples with simulated devices

Useful links:

Dan's intruduction to Bluesky

https://blueskyproject.io/mattermost/

Bluesky originated at National Synchrotron Light Source II

Technical Reports

Bluesky's Ahead: A Multi-Facility Collaboration for an *a la Carte* Software Project for Data Acquisition and Management

Daniel Allan, Thomas Caswell, Stuart Campbell, and Maksim Rakitin
National Synchrotron Light Source II, Brookhaven National Laboratory, Upton, New York, USA

Introduction

To address the growing scale of data at user facilities, a multi-facility collaboration is developing a collection of Python libraries, which are co-developed but may be used *a la carte*, to leverage existing open-source scientific software in general and the scientific Python software ecosystem in particular to improvise cutting-edge experiments and data analysis at the beamline.

The growing velocity and volume of data from third-generation synchrotron light sources and radiation facilities are exposing a data *variety* problem that is particular to user facilities. User facilities manage a large and often changing collection of instruments with a wide span of data rates, structures, and access patterns. Measurements are analyzed with a mix of well-established procedures and improvised techniques. To manage these challenges, user facilities must leverage the growing ecosystem of open-source scientific software, sharing tools and ideas with our counterparts in astronomy, climate science, and particle physics. Tools like GitHub [1], BinderHub [2], and the communities that have arisen around them reduce the friction of distributed collaboration within and between facilities and science domains.

The Bluesky Project is an end-to-end solution [3], encompassing hardware integration [4], experiment specification and orchestration [5], online visualization and analysis, data export [6], and data access [7]. The software is currently in use at all beamlines at the National Synchrotron Light Source II (NSLS-II), a Department of Energy (DOE) Office of Science User Facility located at Brookhaven National Laboratory. It has some adoption at the Advanced Light Source (ALS), another DOE Office of Science User Facility located at Lawrence Berkeley National Laboratory, and is formally planned for wide adoption at the Linear Coherent Light Source II (LCLS-II), located at SLAC National Laboratory, and at the beamlines of the Advanced Photon Source Upgrade (APS-U), located at Argonne National Laboratory—both DOE Office of Science User Facilities. All of the software components have also been employed successfully by an electrical engineering lab [8], working at the lab-bench scale rather than the facility scale, to benchmark hardware performance.

To drive wide collaboration and to overcome "not-invented-here"-ism, the Bluesky Project comprises components with well-defined boundaries that are co-developed but separately useful, and which can be adopted piecemeal. Drawing inspiration from the numpy project, which is the array representation at the core of the scientific Python ecosystem, Bluesky embraces the idea of *protocols* to drive interoperability. As in the case of numpy, protocols enable individually useful parts to be repurposed and extended in ways unforeseen by the original authors. Facilities and groups can meet their own needs and deadlines, while collaborating on a shared core. Figure 1 illustrates the roles and relationships of the project's software libraries.

The data variety problem at user facilities

In a typical experiment at a user facility, data and metadata are scattered. Some critical context for interpreting the data is stored only in people's memories. Metadata may be recorded cryptically in filenames or written in paper notes. Data may be encoded across a variety of formats—some proprietary—which can only be read by specialized software. This approach is manageable up to tens of data sets, but it does not scale well when the number of data sets rises with the data rates from faster detectors and brighter light sources.

What are the specific limitations of the status quo? The data are not machine-readable or searchable, and the relationship between any two pieces of data is unclear. It is difficult to automate away the rote steps of analysis, and code is difficult to generalize or reuse for future experiments. Multimodal analysis, involving plumbing together different techniques with different conventions, is labor-intensive. The traditional approach is also not usually streaming-friendly, which precludes exciting possibilities like observing partial results in real time or performing adaptive experiment steering.

To scale to modern data velocities and volumes, software must systematically track all of the data and metadata necessary for later analysis. This includes:

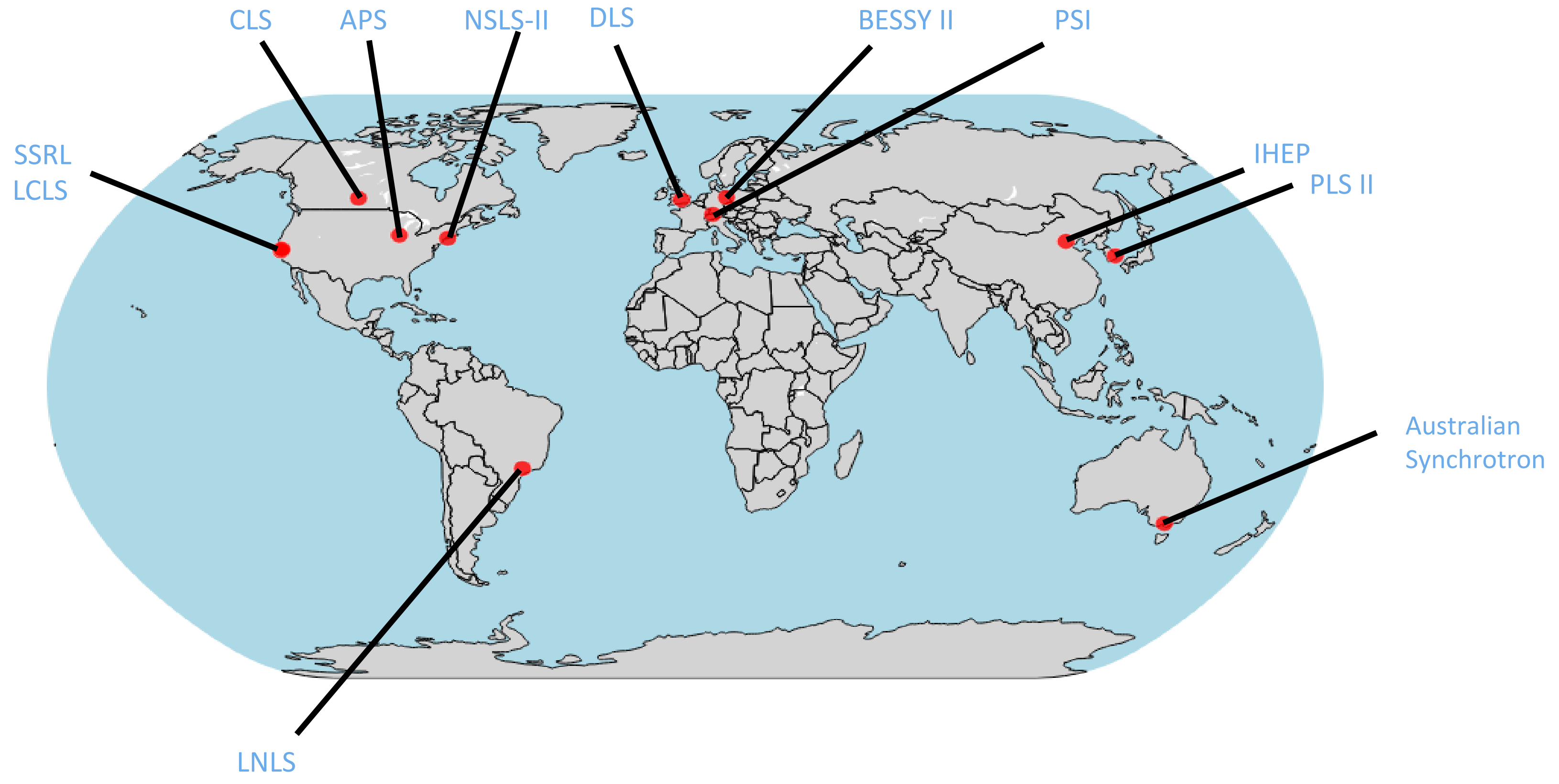
- Timestamps;
- Secondary measurements (e.g., motor positions, sample temperature);
- "Fixed" experimental values (e.g., distance from sample to detector);
- Calibration data;
- Hardware settings (e.g., exposure time);
- Hardware diagnostics;
- Physical details of the hardware (e.g., physical scale of a pixel).

It encompasses information about the sample and the purpose of the measurement, including:

Bluesky's ahead

Collaborative approach

- HPC is becoming more accessible, also thanks to HPCaaS.
- Easier to reach collaborators as calls can be done regularly (share common components).
- Lightweight, portable system-level virtualization became more popular (containers).
- Used by many facilities worldwide, and countless labs.



What’s Bluesky?

- It’s a mini-ecosystem which leaves plenty of room for innovation.
- It uses software design patterns that encourage building on a shared core.

https://blueskyproject.io/

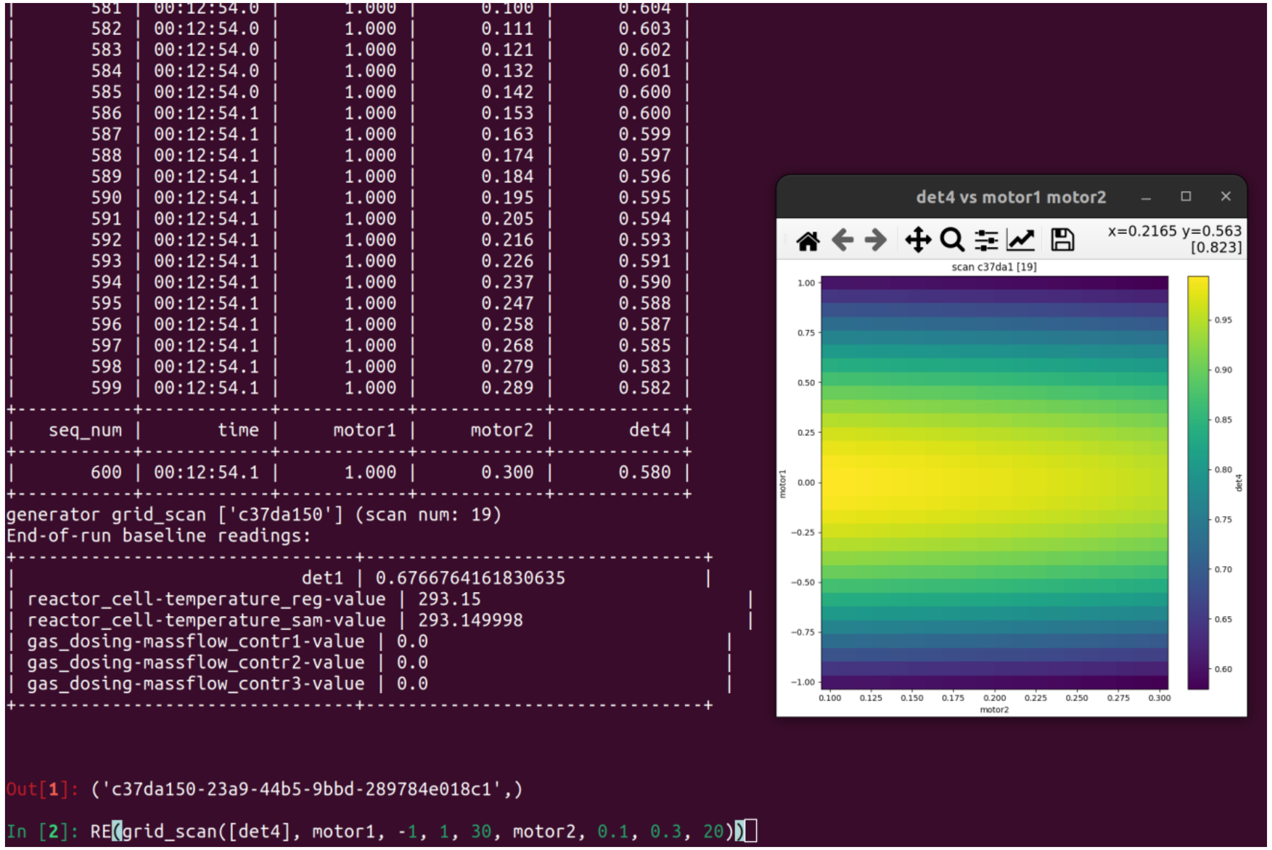
https://mybinder.org/v2/gh/bluesky/tutorials/main

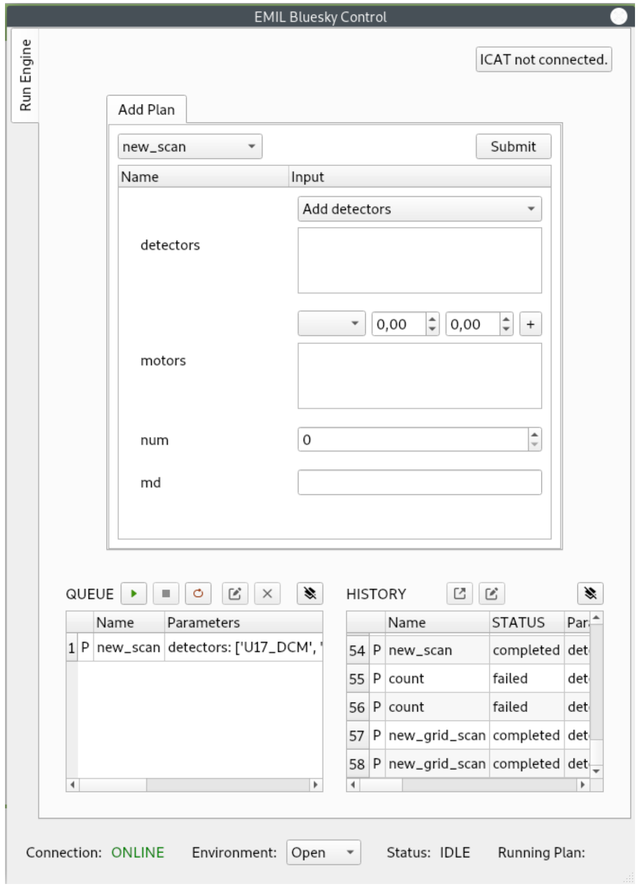
Motivation

- Sources got brighter; detectors got larger and faster: greater data velocity and volume.
  - all the metadata and data has to be tracked (crucial experiment information often not monitored).
- This exposes the variety problem we have at user facilities:
  - Large and changing collection of instruments (often controlled using different frameworks).
  - Wide span of data rates, structures, and access patterns.
  - Mix of well-established data processing procedures and original, improvised techniques.

Motivation

- First target was users coming from SPEC – terminal based interactive workflow. [Bluesky vs SPEC]
- Queue Server with a graphical user interface.

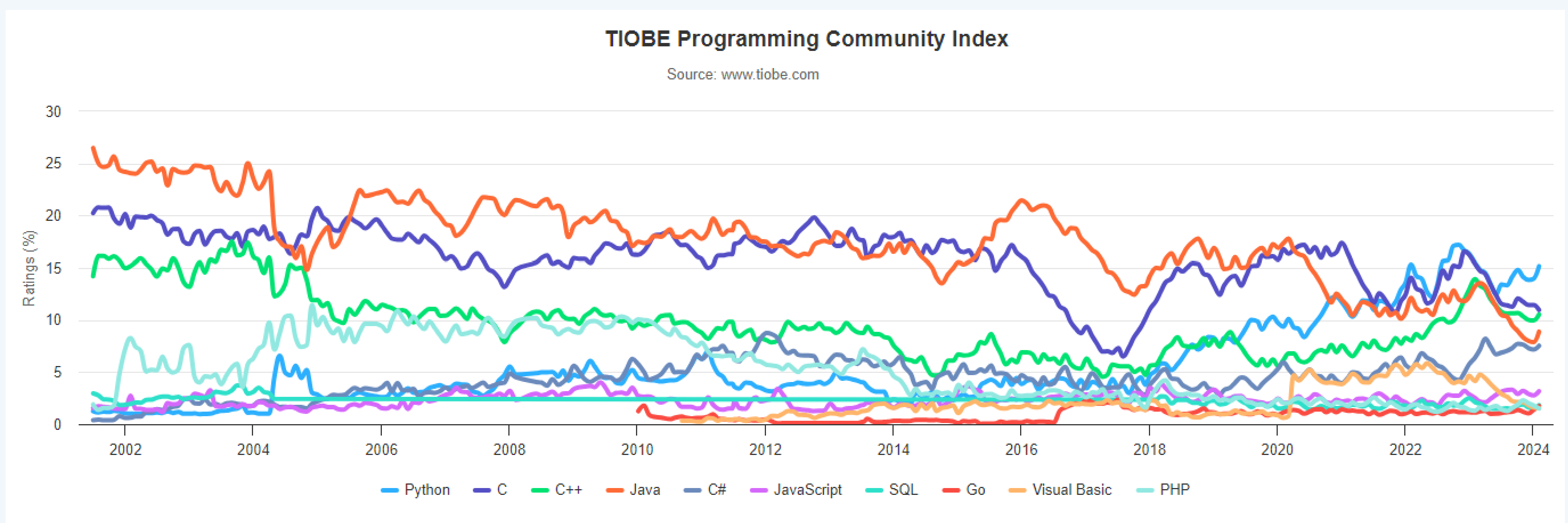




Sebastian Sachse (HZB) GUI

What changed to make data problems easier?

- Each beamline is one of a kind by design, so no software can be a complete solution for everyone.
- Aim to enable beamlines to share yet also support their unique needs.
- Bluesky is python-based and designed in service to data analysis



Bluesky

Free, open-source scientific software exploded.

- Designed to be a set of tools, which can be reused, extended or replaced depending on the specific needs, but it strongly depends on the community