

The *Tiled* Structured Data Service (Now Featuring Awkward Arrays)

Daniel Allan

Data Engineering Group Lead

Data Science and Systems Integration Program

National Synchrotron Light Source II

There is a Broad Effort to Build Data *Services*

- Earthmover: *Arraylake*
- CERN: *Rucio*
- IRIS-HEP: *ServiceX*
- Pangeo: *Pangeo Forge*
- Janelia: *DVID*
- Multiple efforts at Microsoft, NASA, etc.
- Uncountable in-house industry projects

"Today we have lots of data, yes, but it's not universally accessible or understandable. We should invest in data access technologies and metadata standards to allow the world to ask questions without munging."

- Andy Mueller, Scikit-Learn / Microsoft

Source:

<http://matthewrocklin.com/blog/work/2022/07/19/scipy-mission>

Data Services streamline these steps

- **Find** the data
- Access the **specific part** we need (maybe much less than all of it)
- Obtain it in a **format** our data analysis program can read
- Understand what the **names** mean
- Follow **relationships** between data sets
- As needed, provide **open** access or secure, **authorized** access
- Meet the **F.A.I.R.** principles achievable with reasonable effort

File browsers that can't "see inside" the files present barriers to science

- The format data is **written in** (e.g. by a detector) is not necessarily always the most **convenient** or **performant** one.
- **Often, no partial access** ("Just download the first couple points from each of a bunch of files so I can look at them.")
- No clean way to express **relationships between data across files**, especially linking derived data to raw data
- No way to **preview** many scientific formats in a web browser
- **Search** capabilities are limited.

Index of /

Name	Last modified	Size	Description
 folder1/	2014-09-30 22:08	-	
 folder2/	2014-08-06 19:14	-	

Apache/2.4.7 (Ubuntu) Server at localhost Port 80

Materials research has particular challenges

- Large variety of...
 - Shapes (tables, images, image time series, hierarchical structures)
 - Sizes (kilobyte to terabyte)
 - Access patterns
- Highly dynamic requirements
 - Equipment added and removed
 - Evolving structures (framing detectors to event-based detectors)
 - Evolving formats and schema evolution
 - In other domains (astrophysics, climate science, business analytics) the data structure are much more static and defined in advance.

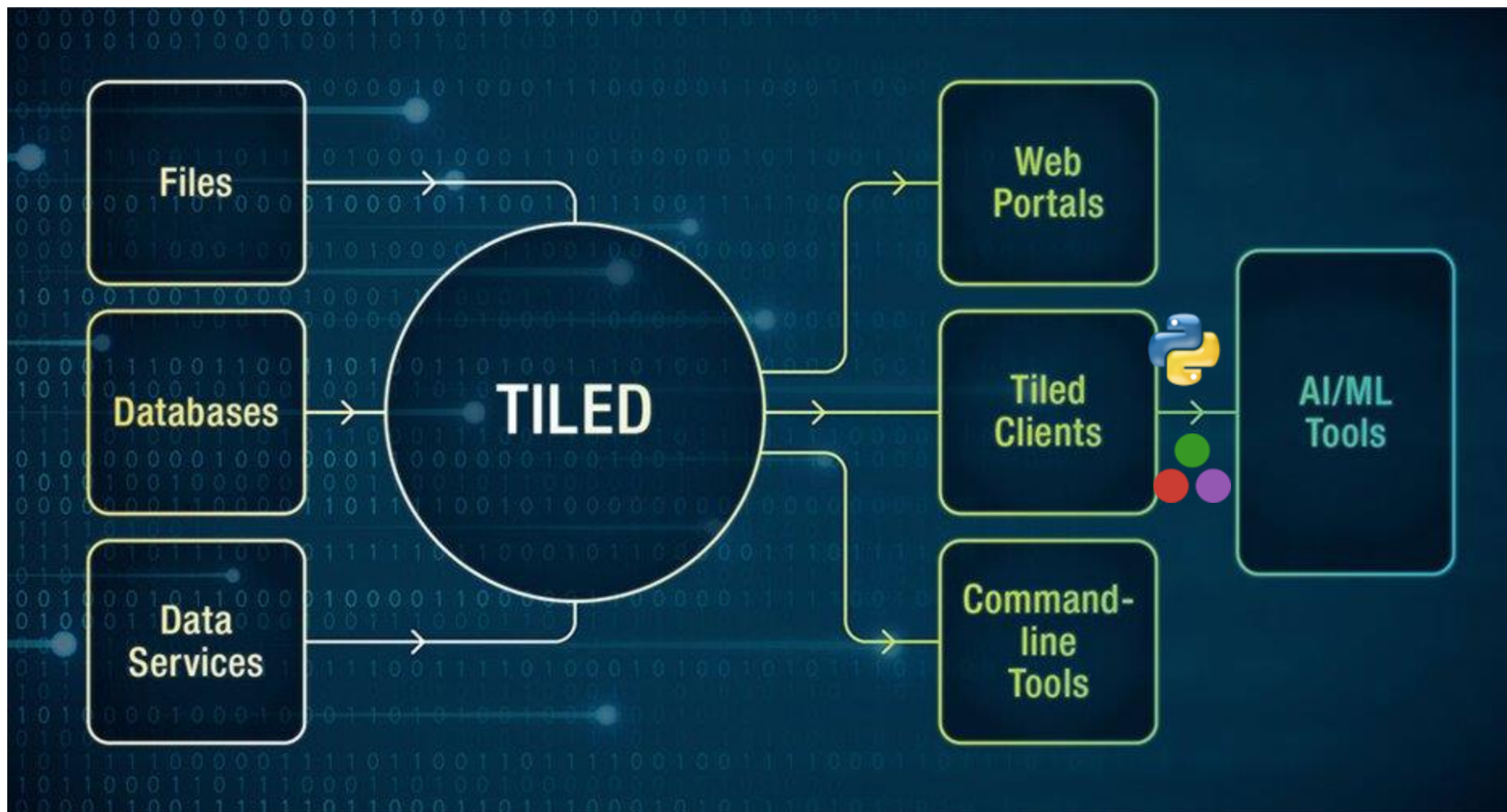
We have built a Data Service called *Tiled*

- Less than 3 years old, still under rapid development
- We have spoken with many, many external groups through the design and development process to better understand this space, our own requirements, and opportunities (including IRIS-HEP 2 years ago)
- *Tiled* passed routine security scans and a *specifically targeted* cybersecurity "pen test" with a clean bill of health
- Soft-launched (open to staff and early users) at NSLS-II

Seeking Feedback from this Group

- What opportunities are we missing in how we frame the problem?
- What related work should we know about or coordinate with?
- Are there any important capabilities we are not considering?
- Are there misfeatures or technology choices we will regret?

Big Picture



Emphasis on a fixed set of well-defined *structures* makes this tractable

- Strided array (like numpy)
- Column-major table (like Apache Arrow)
- Wide, nested structure of these (like a directory or HDF5 group)
- Sparse array (different enough semantics that it gets its own thing)
- Awkward array (nested, variable-sized arrays, *a la* HEP ROOT)
 - We are just beginning to use event-based detectors at synchrotrons
 - We are interested in using Awkward / ROOT examples to drive big-scale performance tests while the project is still malleable

Key Requirements, Features

- *Structured* access to data
 - Request a slice of a large array, a subset of columns in a table, etc.
- Transcoding and compression
 - Be unopinionated about data formats: meet users where they are
 - Different applications need different formats (think data analysis vs web viz)
- Fast search on metadata, backed by a SQL database
 - PostgreSQL or SQLite: extremely robust, trusted, well-understood technology
 - These now support indexed JSON columns, like a document database, for fast search on sometimes-messy experimental metadata
- Access control, using web security standards
 - Integrates with BNL authentication
 - Integrates with NSLS-II proposal system to determine authorization rules

Two Modes of Use

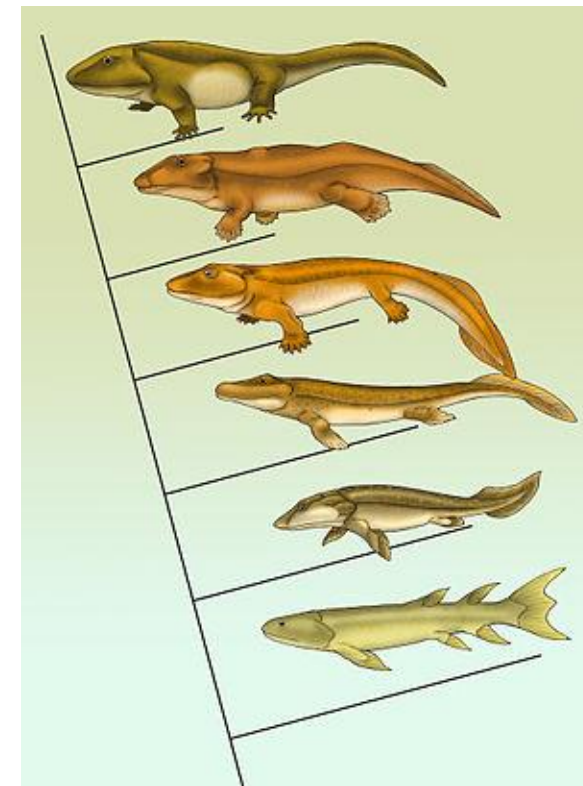
- Tiled as active source of truth
 - HTTP clients write metadata and data into Tiled (in whatever format)
 - Tiled stores it in an opinionated format
 - SQL + files is the source of truth, "managed" by Tiled
- Tiled as passive cache
 - Read-only view of existing data
 - Take formats as they are, index file metadata, structure, location in SQL
 - Coexists peacefully with existing file-based workflow on those files
 - Cache could be blown away and recreated perfectly

***Tiled* incorporates the input and experience of many people**

- **NSLS-II**: Dan Allan, Stuart Campbell, Thomas Caswell, Marcus Hanwell, Juan Marulanda
- **ALS**: Dylan McReynolds, Joseph Kleinhenz
- Builds on open-source collaboration with Martin Durant (**Anaconda, Inc.**) with involvement from Garrett Bischof (**NSLS-II**).
- Recent contributions from the **IRIS-HEP** collaboration to add support for AwkwardArray

Wading out of the Data Lake

0. Heap of bytes (at least you *have* the data...)
1. A database of filepaths (or URIs) with associated metadata would let us *search*.
2. But if the server knows enough about the files to *open* and *read* them and describe their *structure* to clients, clients don't know have to know the storage details to slice into the data structurally.
3. And, for extra credit, if the markup follows some convention that we can communicate to the client (e.g. "XDI" or "NXcanSAS") we can go further still.



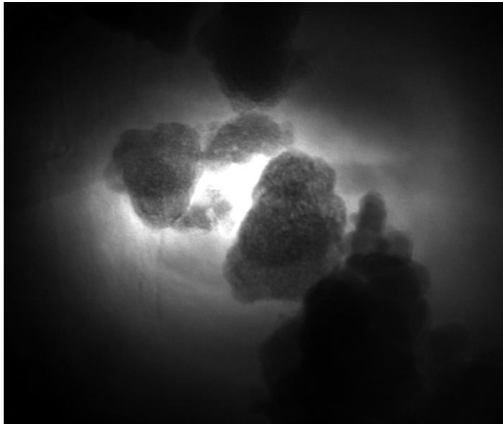
Data in a web browser, on a phone

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 44 22

0 19 10

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 44 22

0 19 10

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!

Like Jupyter Notebook, Tiled is something a single user can easily run

- Easy to run on a laptop:
 1. `pip install tiled[all]`
 2. `tiled serve directory my_files/`
 3. Access data from your web browser or Python or any Internet-connected program.
- Tiled can also scale up for large **multi-user deployments**, like JupyterHub.

Support Infrastructure

- The project is 3-clause BSD licensed and covered by the Bluesky Project multi-facility governance model github.com/bluesky/governance
- It is supported as an open source project by NSLS-II's Data Science and Systems Integration Program.
- NSLS-II deploys it on the public web (tiled.nsls2.bnl.gov)
- **At NSLS-II we are betting our data access on this.**

Seeking Feedback from This Group

- What opportunities are we missing in how we frame the problem?
- What related work should we know about or coordinate with?
- Are there any important capabilities we are not considering?
- Are there misfeatures or technology choices we will regret?



Demo

Links

Demo: <https://tiled-demo.blueskyproject.io/>

Documentation: <https://blueskyproject.io/tiled>

Code: <https://github.com/bluesky/tiled>

Contact: Daniel Allan <dallan@bnl.gov>