HZB BESSY II Light Source

PROJECT STATUS REPORT

Teresia Olsson,
on behalf of the pyAML steering committee

3rd Python Accelerator Middle Layer Workshop, 2-4 February 2026

pyAML

CONTENT

- Recap of what has happened since the last workshop.

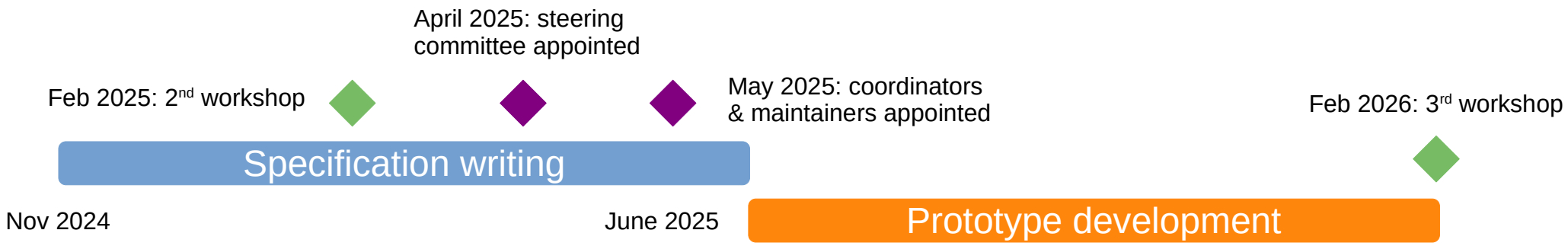
- But also focus on:
  - Lessons learned during the process.
  - Looking ahead towards the next steps and things to discuss for the future.

- Also aim to put our collaboration into the broader context of research software development in accelerator physics → start conversion about our niche and role in the community.

WHERE ARE WE AT THE MOMENT?

- Currently in the prototype phase: started at last workshop → ends at this workshop.

- Goal: develop a prototype for the software to be evaluated by the community.
  - Write a software specification.
  - Implement a few use cases (tune correction & orbit correction).
  - Explore different options and test them.

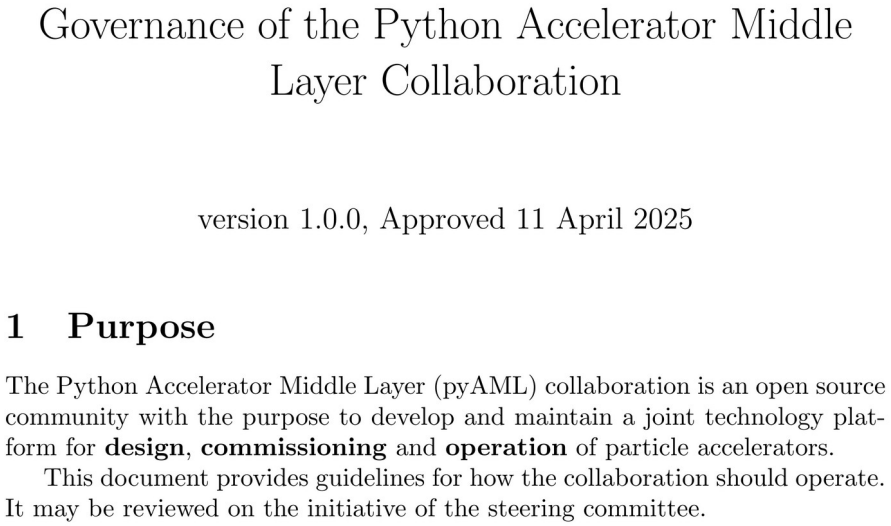


- Has also been a period to develop the collaboration itself and figure out how it should be run.

ORGANISATION

GOVERNANCE

- Discussions started at last workshop → document approved by community in April 2025.
- Intentionally kept brief with in mind to extend later if required.



Governance of the Python Accelerator Middle Layer Collaboration

version 1.0.0, Approved 11 April 2025

1 Purpose

The Python Accelerator Middle Layer (pyAML) collaboration is an open source community with the purpose to develop and maintain a joint technology platform for **design**, **commissioning** and **operation** of particle accelerators.

This document provides guidelines for how the collaboration should operate. It may be reviewed on the initiative of the steering committee.

- Steering committee appointed at same time → based on 5 labs which committed most FTEs.
- Appointed for 1 year → plans for the next steering committee needs to be discussed soon.

Current steering committee

Facility	Member	Deputy
DESY	Ilya Agapov	Konstantinos Paraschou
ESRF	Simon White	Simone Liuzzo
HZB	Teresia Olsson (chair)	Markus Ries
MAX IV	Marco Apollonio	Stephen Molloy
SOLEIL	Laurent Nadolski	Patrick Madela

VISION

Excerpt from "Governance of the Python Accelerator Middle Layer Collaboration, version 1.0.0"

- *"The Python Accelerator Middle Layer (pyAML) collaboration is an open source community with the purpose to develop and maintain a **joint technology platform for design, commissioning and operation of particle accelerators**".*

- Three guiding principles:

1. **We have collaboration as a core-value.**

   We aim to develop a **diverse community of people from accelerator laboratories around the world** cultivating an environment of inclusivity, cooperation, and sharing.

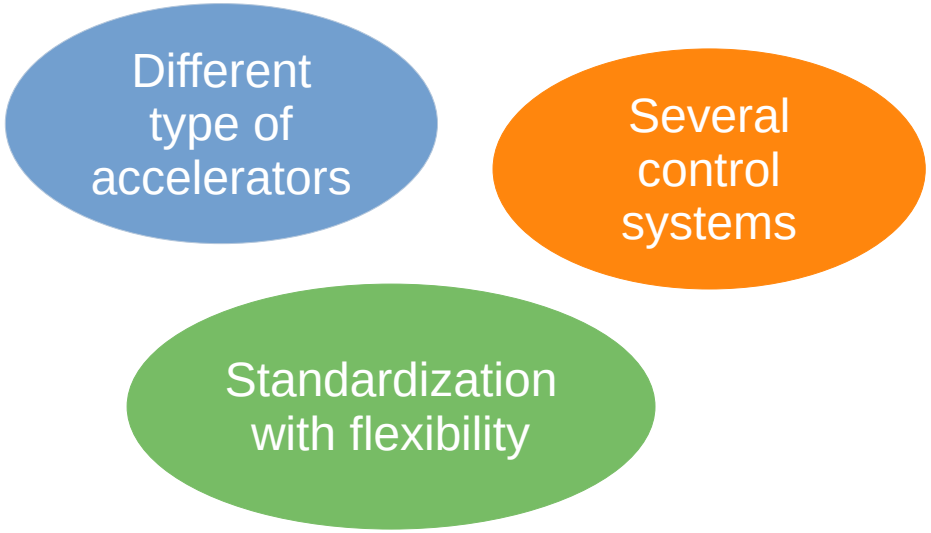
2. **We aim for user-centered design.**

   We will enable **users of all levels** to use the software and participate in the project by keeping barriers low. The software should follow the principles and practices of modern software engineering, be **easy to install** on the systems used by the community, **easy to configure** and **easy to start to use**. We should have a modular architecture which allows users to configure, use and develop algorithms and tools.

3. **We value our data.**

   Configuration data should be separated from source code, easily updated and follow a **common standard** which can be extended if required. We should have a standard for how to save measurement and analysis data together with appropriate metadata.

WHAT DOES THE VISION MEAN?



- Started with purpose to replace Matlab Middle Layer → has developed into something larger.
- Many other projects going on but focused on individual facilities or control system → our vision unique in the community.

- We also have a training component → use the collaboration to increase the software skills among accelerator physicists.
- We are **not** a project where software engineers should deliver a product to be used by physicists → physicists are active developers and software engineers and physicists work closely together.

The ambitious vision comes with challenges → but also makes it exciting!

SPECIFICATION

SOFTWARE SPECIFICATION

Documents available at:
https://python-accelerator-middle-layer.github.io/documents/

- Started by writing a software specification.
- This was a difficult process:
  - Facilities with different requirements and approaches for how they run their machines.
  - Different visions for how the software should work and what type of tool we need.
  - Different views between software engineers and physicists on how a specification should look like.
  - Too few people with experience of writing software specifications in a research software context.
- **Lesson learned:** writing good specification for research software is complicated.

- But now have two documents: **requirement specification** & **user interface specification**
- Both are still drafts.
- Steering committee decided not spend resources on finalizing them, but keep as guidelines and focus on implementation.

MOST IMPORTANT QUALITY ATTRIBUTES

Classification according to Bass, Clements & Kazman, “Software Architecture in Practice, 4th Edition, Pearson Education, 2022.

- Usability
  - Accelerator physicists must be able to use, develop, debug and maintain.
  - Usable for wide range of users (bachelor students to experienced senior scientists).
  - Support the research process: both easy scripting and running complex, standardized applications.
- Modifiability
  - Easy to add new functionality over time.
  - Needs to evolve together with the evolution of accelerator design & technology.
  - Possible to add facility specific functionality.
- Integrability
  - Possible to integrate with existing hardware/control system tools.
  - Leverage the scientific Python ecosystem.
  - Easy integration with HPC and data management resources (clusters, databases etc).

AGNOSTIC INTERFACE

- Support different control systems: DOOCS, EPICS (CA/PVA) and TANGO.
- Possible to write generic devices and the choice of control system is just configuration.

- Same interface to different backends:
  - Live accelerator
  - Virtual accelerator: control system + physics model
  - Simulator: physics model only

→ Develop + test applications using one backend which works without modifications for another backend.

MACHINE INDEPENDENCE

- Independent of facility specific naming conventions → fully configurable.
- Physicists must be able to interact with machine using the names they know → hide control system names from the user.

- Possible to use different types of accelerators (storage rings, ramped machines, transfer lines, linacs etc.)
- Default simulator will be pyAT (python accelerator toolbox) but not best choice for all types of accelerators → in the future be possible to extend to other simulators.

→ Configuration layer crucial for use at different facilities.

CONVERSIONS

Classification according to Y. Hidaka, “Particle Accelerator MIddle LAyer (PAMILA)”, presentation at pyAML community meeting 15 November 2024.

- Two type of conversions:
  - Universal (intra-dimensional) conversion:
    - Example: mA ↔ A, mm ↔ m
    - Handle control systems configured for different units.
  - Representation (inter-dimensional) conversion (often called hardware to physics conversions):
    - Example: A ↔ mrad
    - Handle conversions between different backends but also preference of the user → physicists must be able to use the units they know their machine in.

- Needs to be able to handle combined function magnets (m x n mapping) → integration with magnet excitation curves or models.

→ Flexible and customisable two-way conversion interface.

APPLICATIONS

- Possible to write generic applications that work out of the box after configuration.
- Prerequisites:
  - Grouping of devices to use together: e. g. all horizontal correctors grouped together (often called family).
  - Standardised names: e. g. correctors used for orbit correction mapped to the name OrbitCorrectors.
  - Execution should be controlled at device level → no sleeps in applications → when hardware upgraded instant benefit.
- Modular: measurement and analysis independent.
- Metadata automatically generated from devices and configuration.

→ Sufficient abstraction required to write generic applications that can be shared between facilities.

REALISATION

DEVELOPMENT ORGANISATION

Meeting agendas and minutes available at:
https://github.com/python-accelerator-middle-layer/governance

Maintainers: 21 people

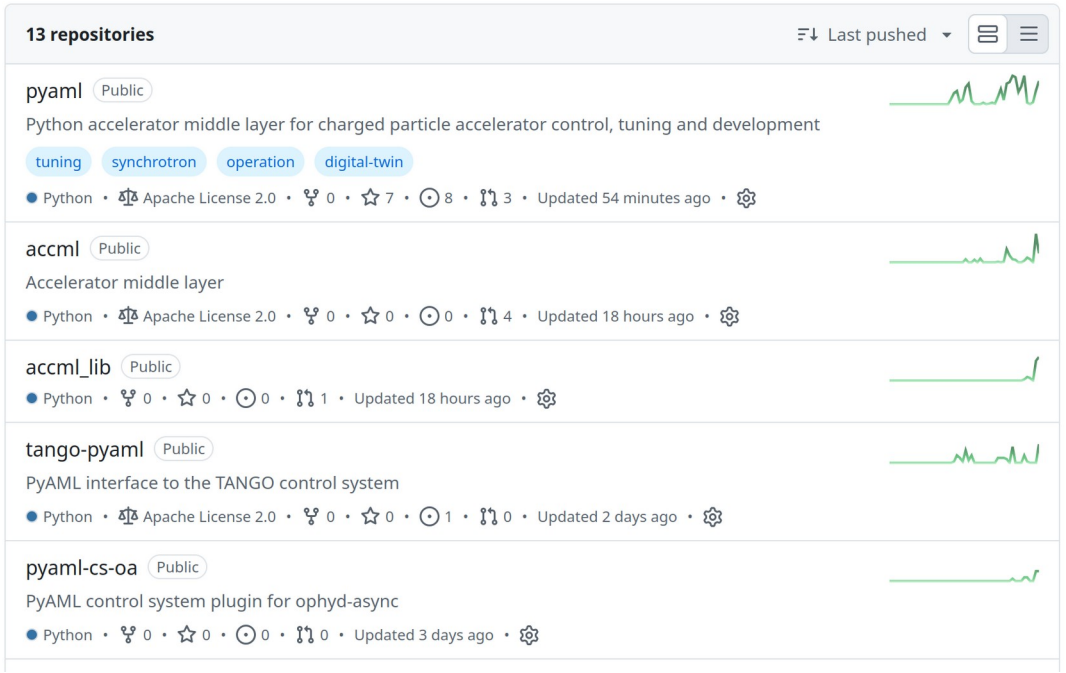
- Appointed by steering committee
- Everyone is welcome to become a maintainer
- Meet every second week + email list

- Manage the GitHub organisation and repositories
- Write code
- Approve pull requests
- Ensuring code aligns with scope and architecture + long-term functionality

Coordinators
and working group leaders:

Vadim Gubaidulin,
Jean-Luc Pons,
Waheedullah Sulaiman Khail

- Code is hosted on GitHub.
- GitHub discussions used as main communication channel to include everyone in the community.

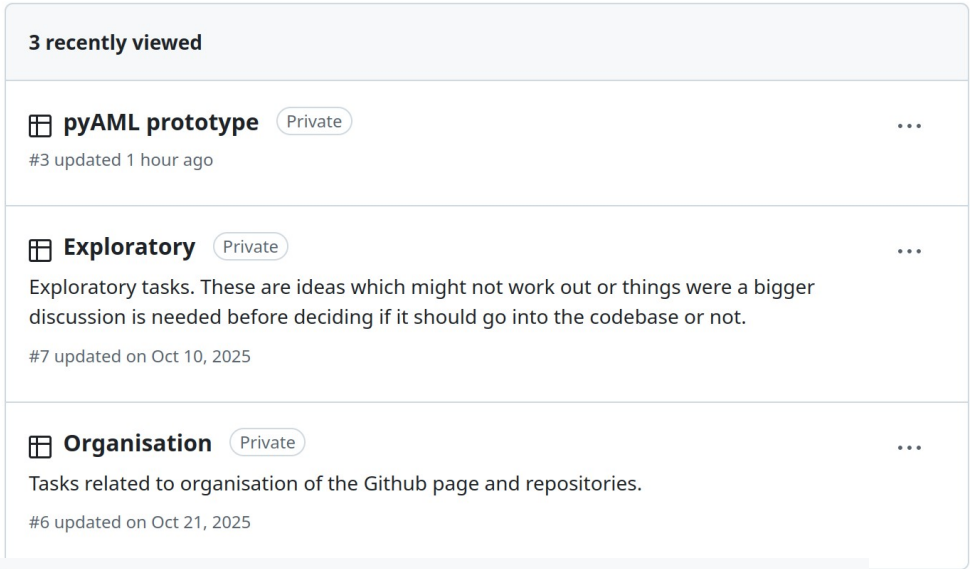


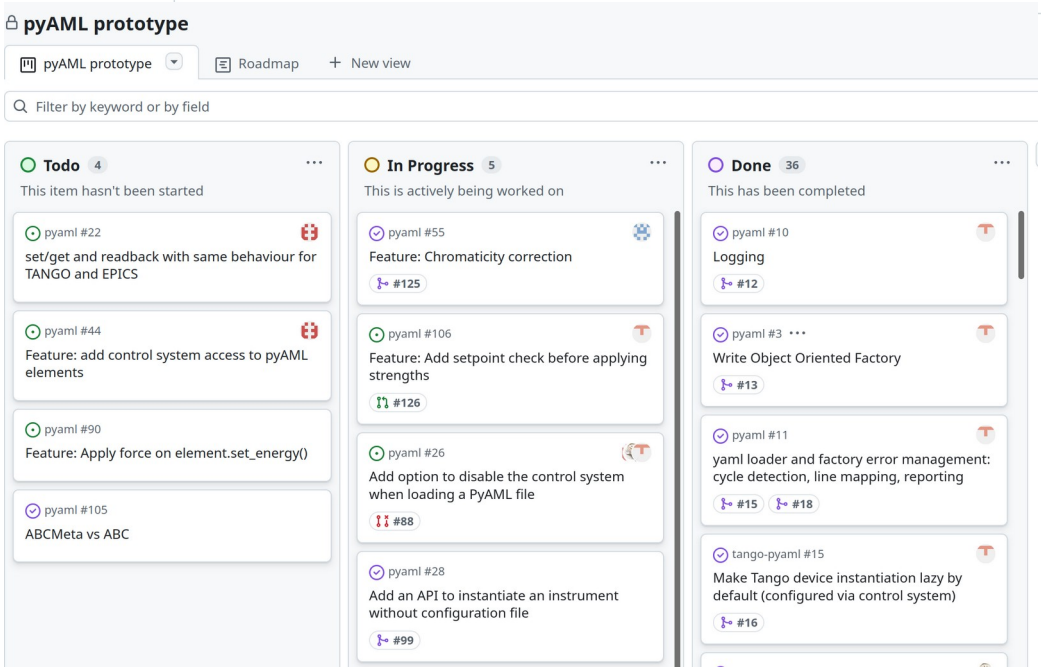
DEVELOPMENT ORGANISATION

- Started to use GitHub projects for project management.
- Tests at ESRF, HZB, MAX IV, SOLEIL.

- **Challenge:** number of active maintainers.
- Number of contributors: 10 people.
- Decided to have two working groups for the prototype phase (core & testing) → has not really worked like that.

- What should the organisation be moving forward? → Working groups and coordinators were only decided for the prototype phase.





FUNDING

Explored funding opportunities

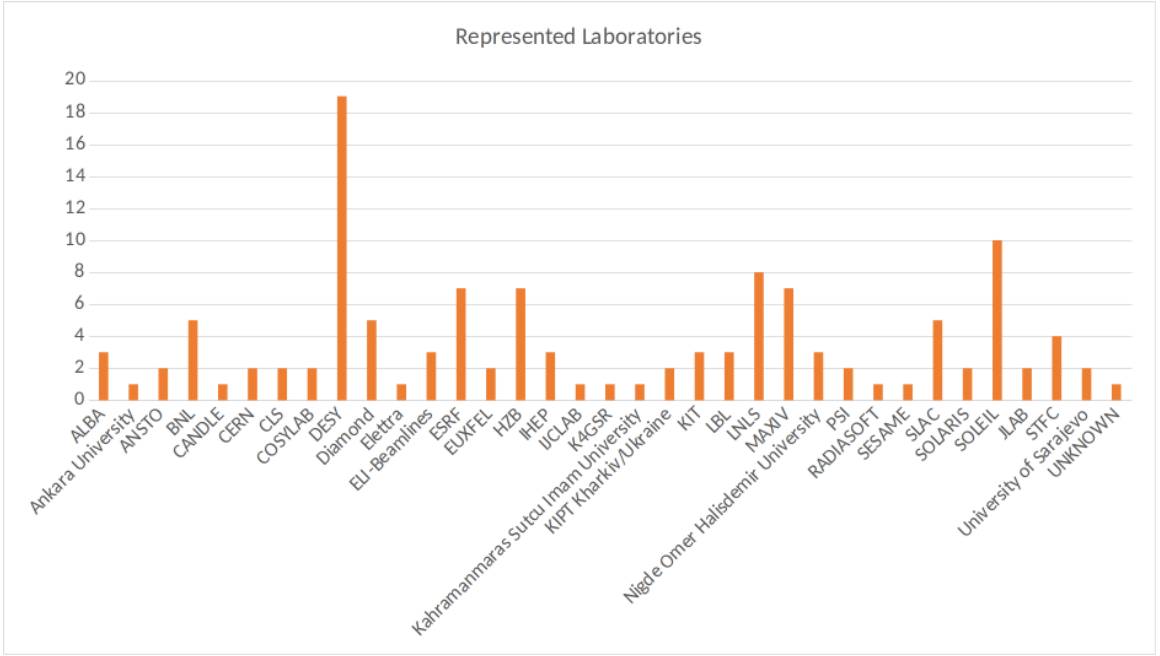
Call	Focus	Status
COST action (European Cooperation in Science & Technology)	Networking and cooperation	**Not funded**
HMC project call (Helmholtz Metadata Collaboration)	Metadata and standardization	**Not funded**
LEAPS (League of European Accelerator-based Photon Sources)	Promote activities of importance for LEAPS members	**We have been included in the roadmap**

- LEAPS will continue to support our activities (such as workshops and trainings):
  - We will be involved in applications for EU funds when possible.
  - We can mention being part of the LEAPS roadmap in future applications for funding.
- Main source of resources currently is FTEs committed by facilities.
- But no formal agreement about FTE contributions → voluntarily, best-effort contributions.

COMMUNITY

- Status email list January 2026:
  124 people, 34 organisations, 21 countries

Status email list January 2026



- Community meetings every 3rd month.
- We actively reach out to speakers from different type of facilities around the world + take care to include speakers outside maintainers' group.
- Developed into a forum for sharing work around hardware abstraction, experiment orchestration, high level applications etc in the accelerator physics community → has its own value since such a forum has somewhat been missing.

- Actively reaching out to other communities and build connections (e. g. the Particle Accelerator Lattice Standard (PALS) project and the Bluesky community).

OUTLOOK

CHALLENGES

1. Our ambitious vision

- Our community members are diverse with different needs and visions.
- Many already have existing projects and tools → rather than replace this they want to integrate with pyAML.

2. Software skills

- We only have a few software engineers involved in the project + lacking EPICS control system engineers.
- Varying software skills among the accelerator physicists → a big Matlab community in process to migrate to Python.

3. Resources

- No dedicated funding that the collaboration can decide over itself.
- All FTE contributions are currently from the ring-based light source community → but this community is at the moment already pressed for resources due to facility upgrades.

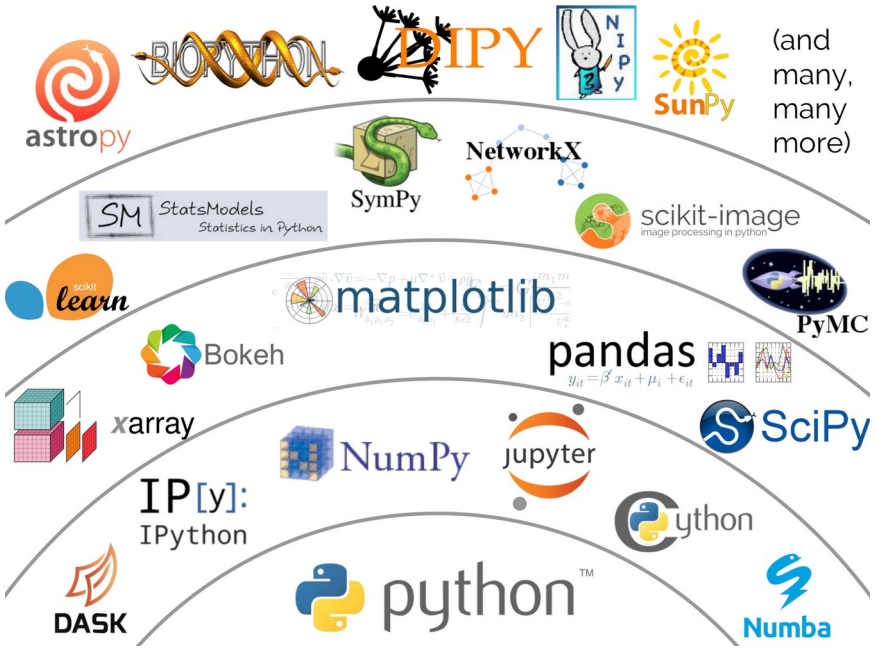
OPPORTUNITIES

- Looking at similar software collaborations
  (examples: Astropy, scikit-HEP, preCICE, Bluesky project etc.)
  → develop following ecosystem approach.

- What does it mean?
  - Not one package but a collection of packages compatible with each other where the user can pick and choose which parts they want to use.
  - Require high modularity and clear interfaces → coordination and versioning essential.
  - Allow for several different implementations of the same functionality → this we have already started with control system bindings.

- Briefly discussed by the steering committee as one possibility for us → community discussion and input required during this workshop for roadmap decision.

The scientific python ecosystem



Credit: Jake vanderPlas, "The Unexpected Effectiveness of Python in Science", PyCon 2017

THANK YOU!

For updates see the webpage:
https://python-accelerator-middle-layer.github.io/