



NEURODATA
WITHOUT BORDERS

The Neurodata Without Borders Ecosystem for FAIR Neurophysiology Data

Driving Collaboration in Neuroscience

Oliver Rübel

Computational Biosciences Group
Scientific Data Division
Lawrence Berkeley National Laboratory



BERKELEY LAB
Bringing Science Solutions to the World

Contents

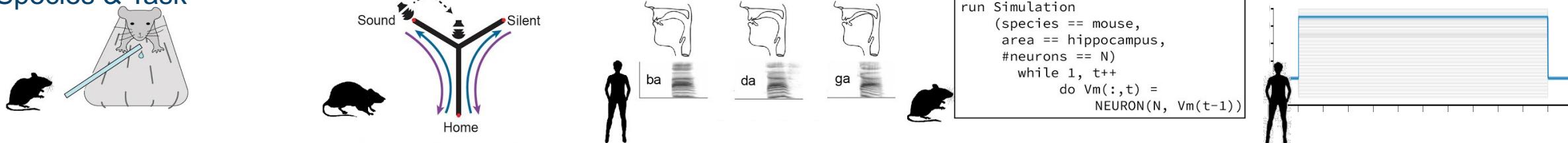
- Motivation
- What is NWB?
- NWB Data Analysis, Visualization, and Management Tools
- Core NWB Data Standard Ecosystem
- The Energy Barrier in Data Standardization

Motivation:

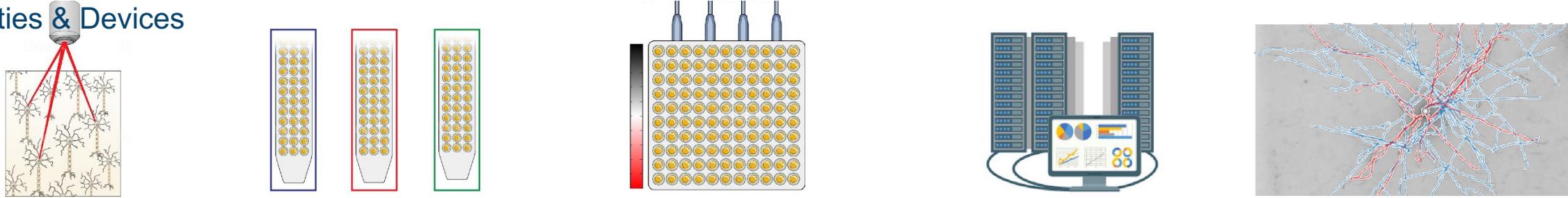
- Data are expensive to collect
 - Neuroscience datasets are often rich and can be used to answer many different questions
 - Reusing data saves money and time and reduces animal use
- Sharing neurophysiology data within a lab and with collaborators is tedious
 - Everyone has a different way of storing their data
 - Metadata required for data interpretation is often missing or stored ad-hoc
- Scientific results and analyses are hard to reproduce and compare
 - Processing, analyzing, and visualizing neurophysiology data requires converting data from one esoteric format to another
 - Metadata is inconsistent and often missing

Challenge: Diversity of experiments and rate of innovation

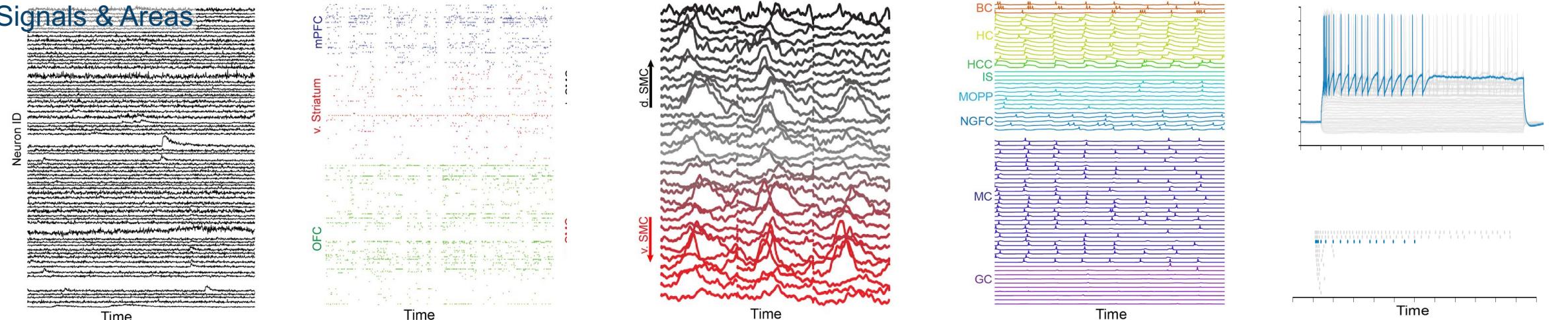
Species & Task



Modalities & Devices



Signals & Areas

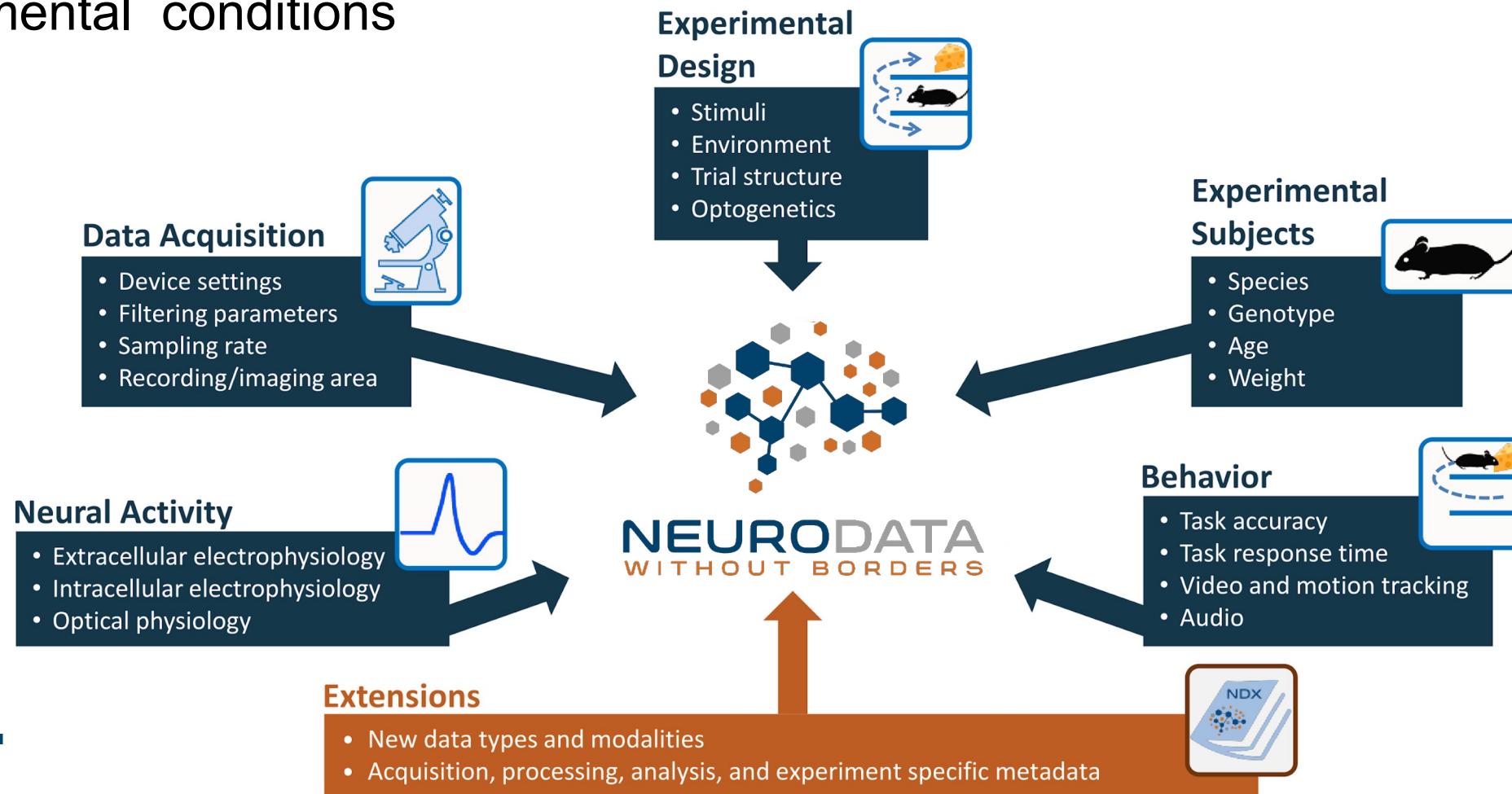


What is NWB?

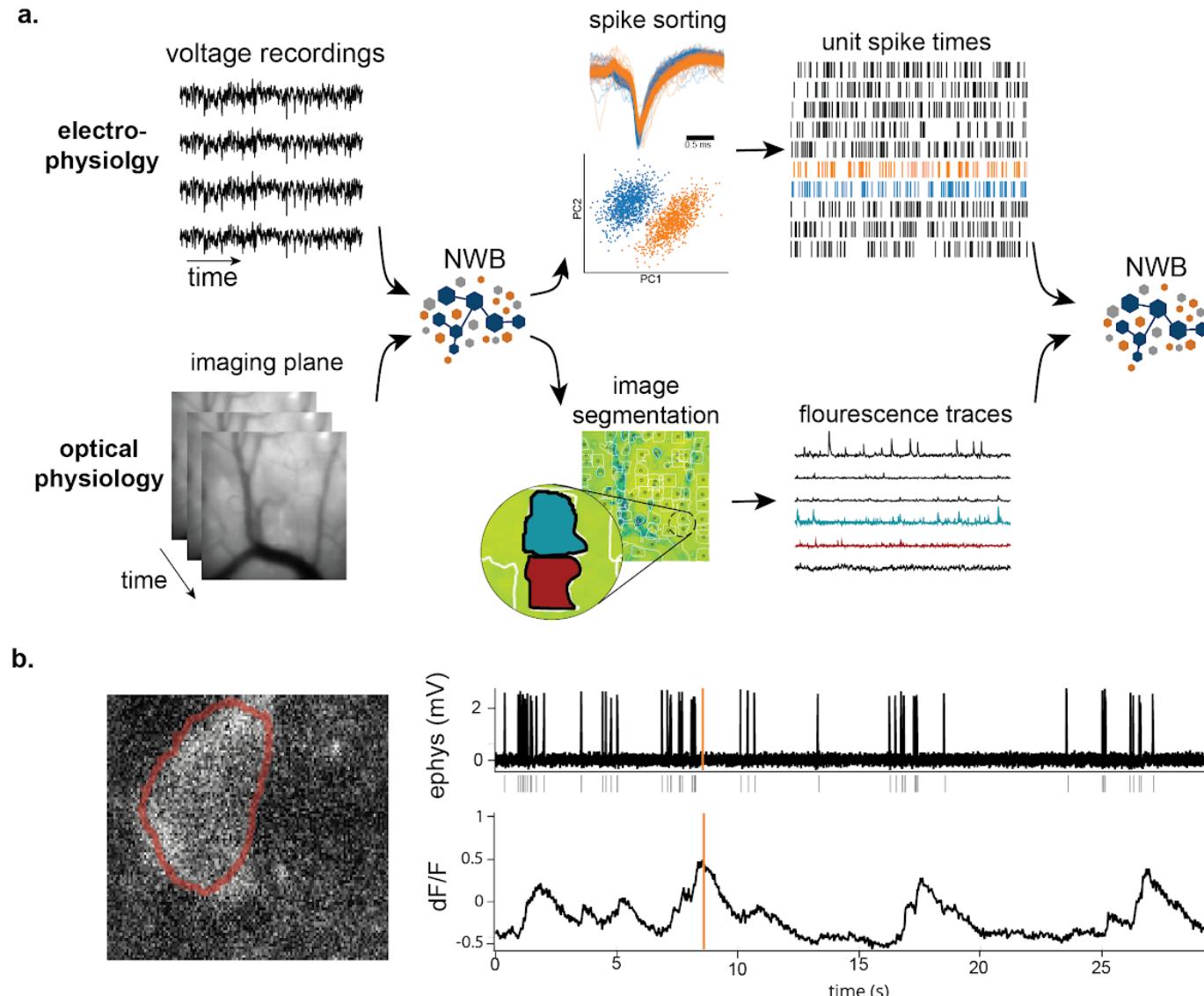
An Ecosystem for Neuroscience Data Standardization

A unified data standard for neurophysiology

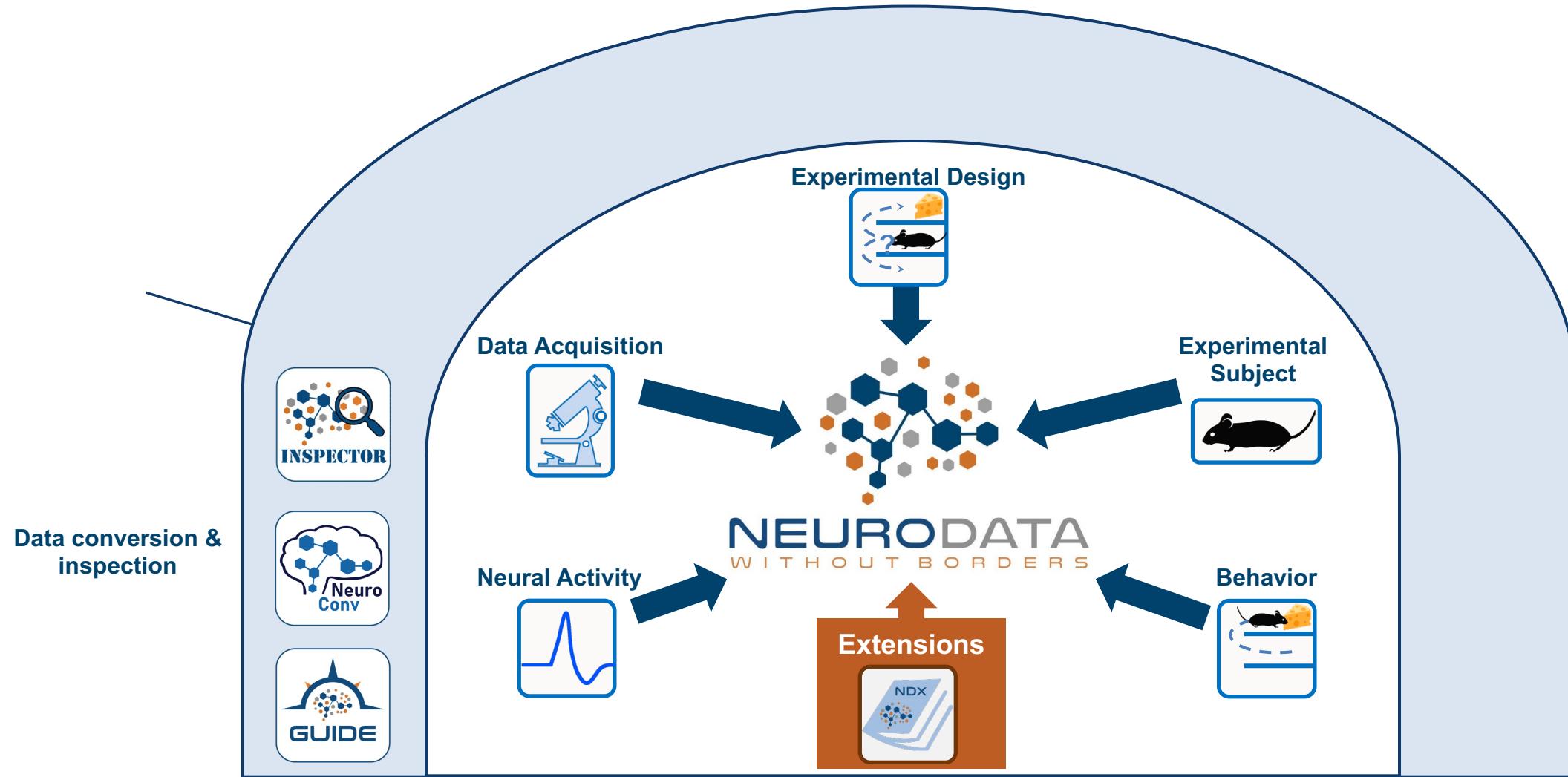
The NWB data standard defines a unified data format for neurophysiology data, focused on the dynamics of groups of neurons measured under a large range of experimental conditions



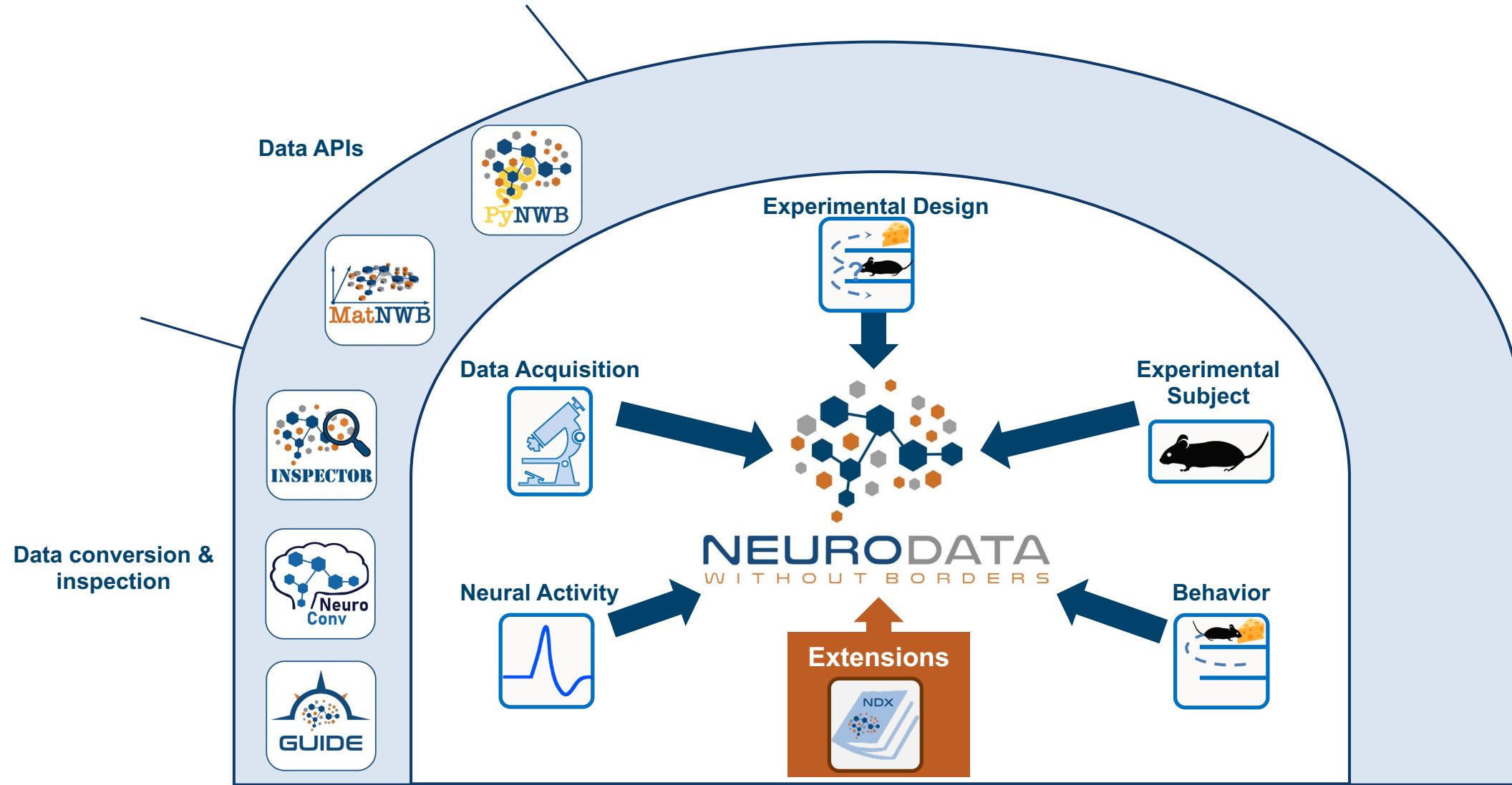
NWB enables unified storage of multimodal raw and processed data



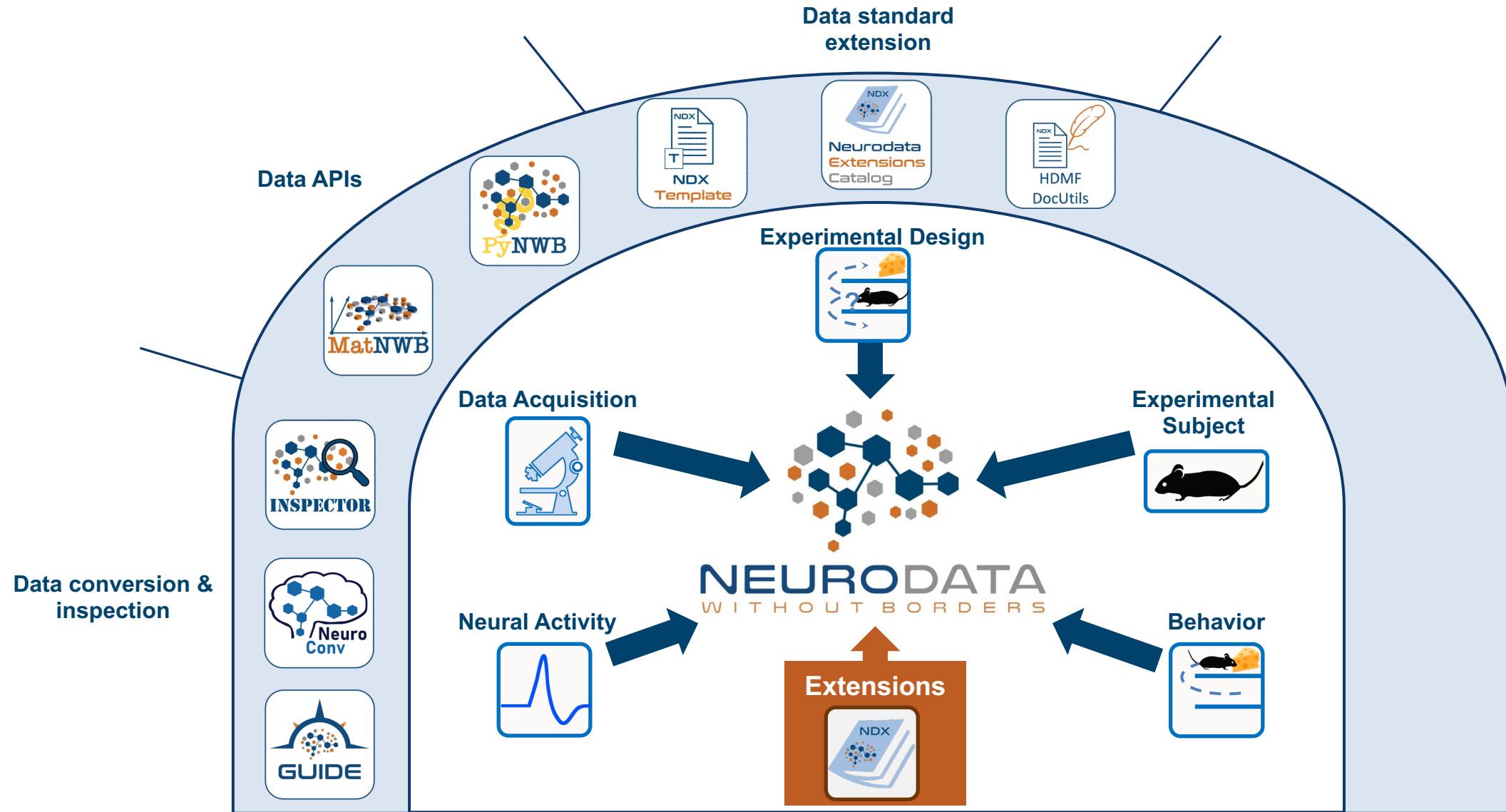
An ecosystem for neuroscience data standardization



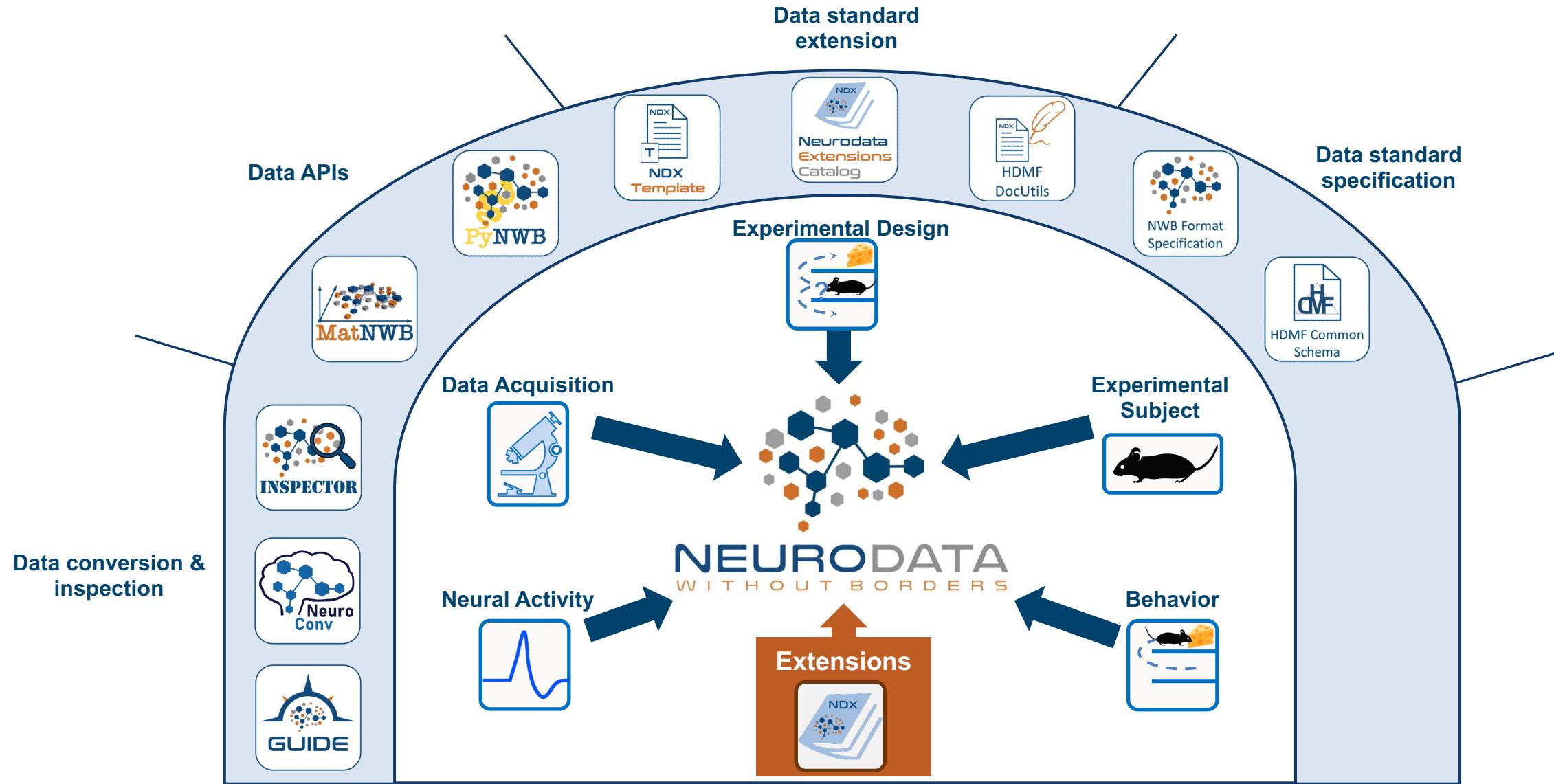
An ecosystem for neuroscience data standardization



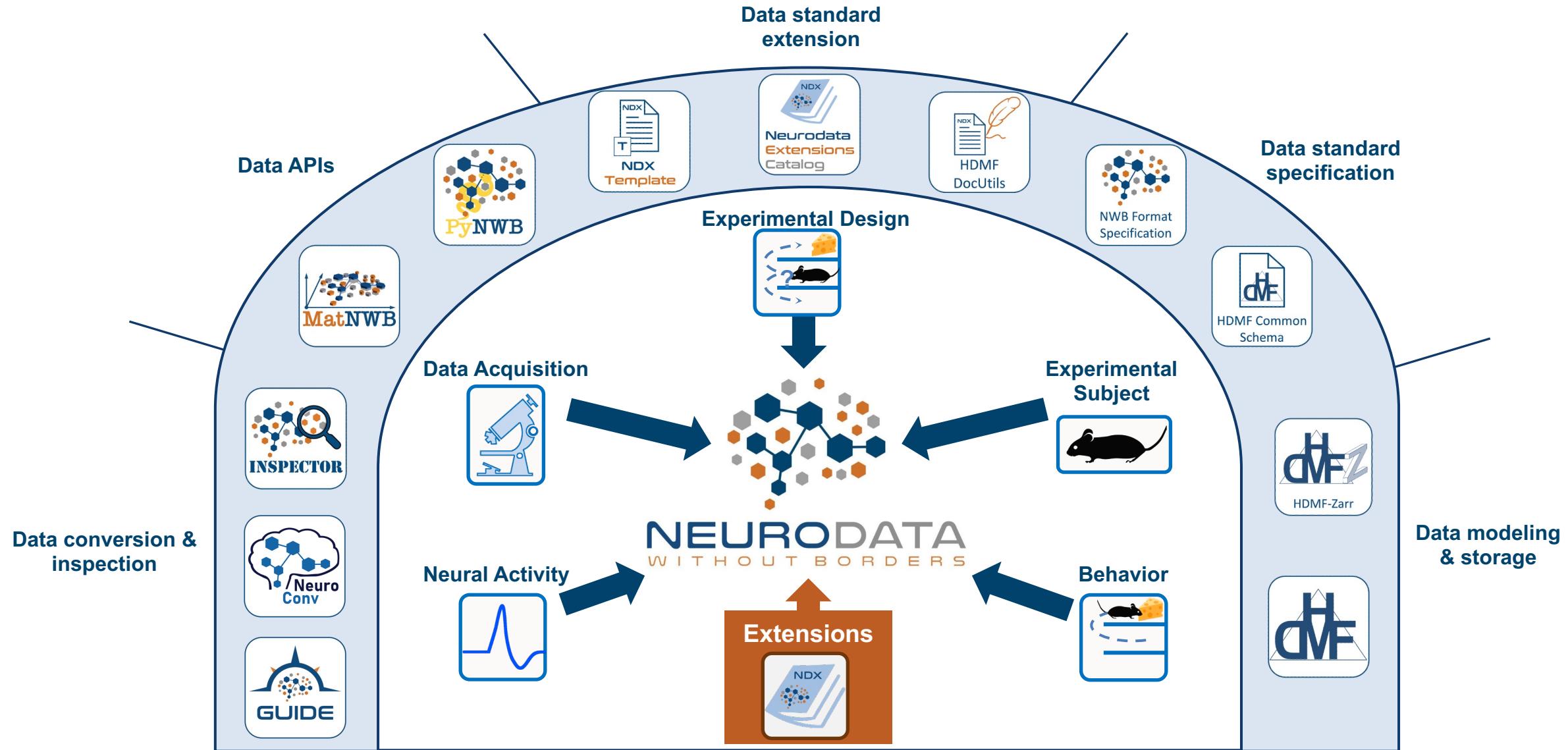
An ecosystem for neuroscience data standardization



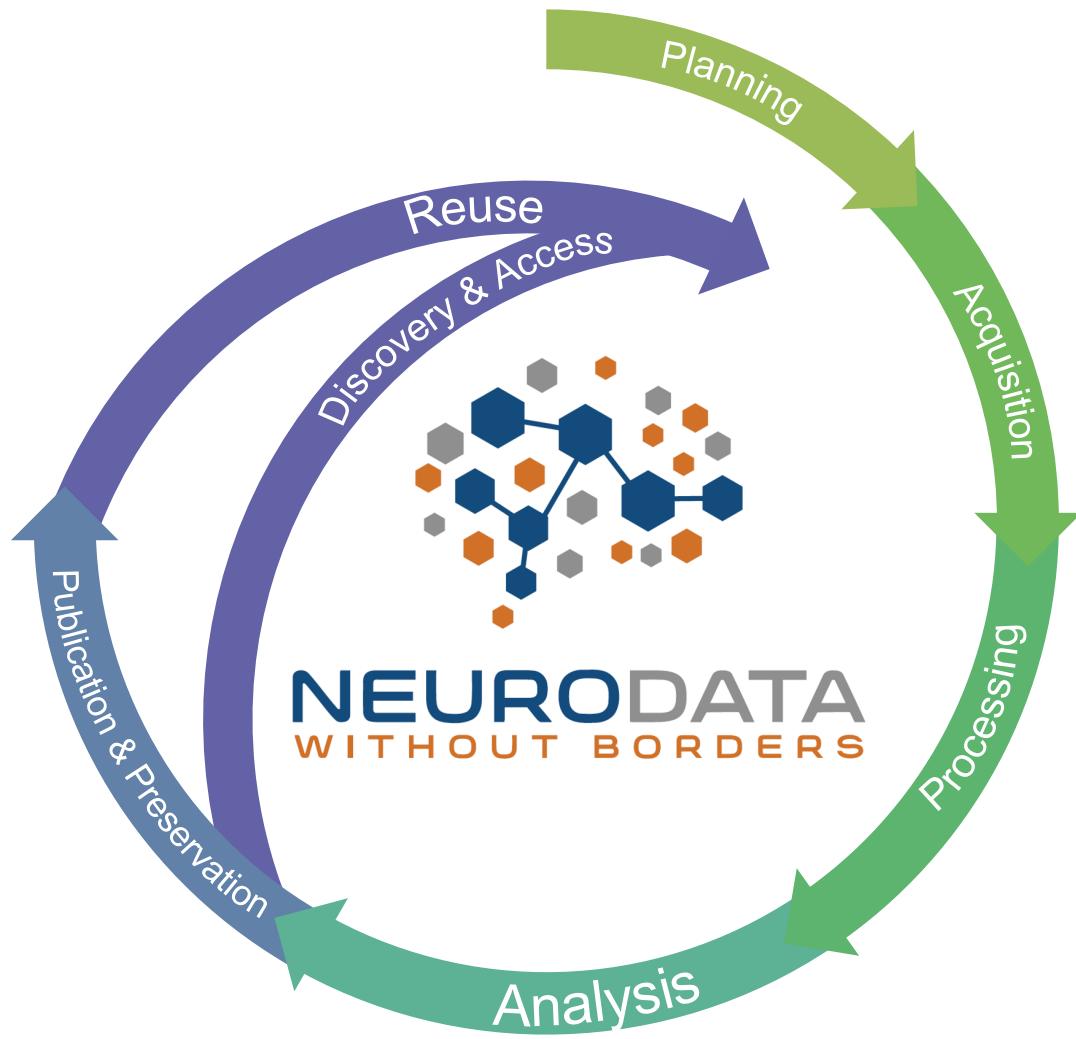
An ecosystem for neuroscience data standardization



An ecosystem for neuroscience data standardization



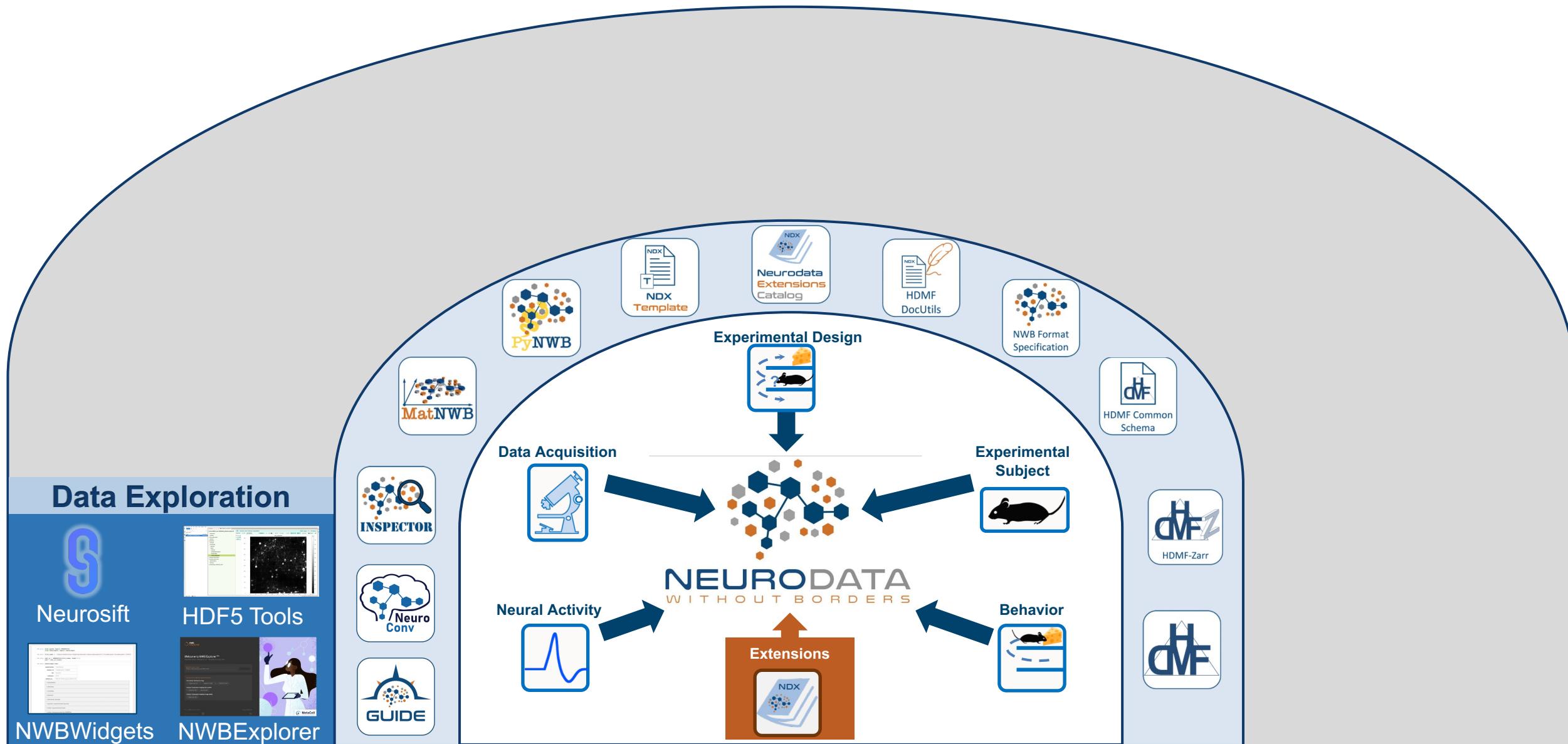
NWB technologies at the heart of the neurodata lifecycle



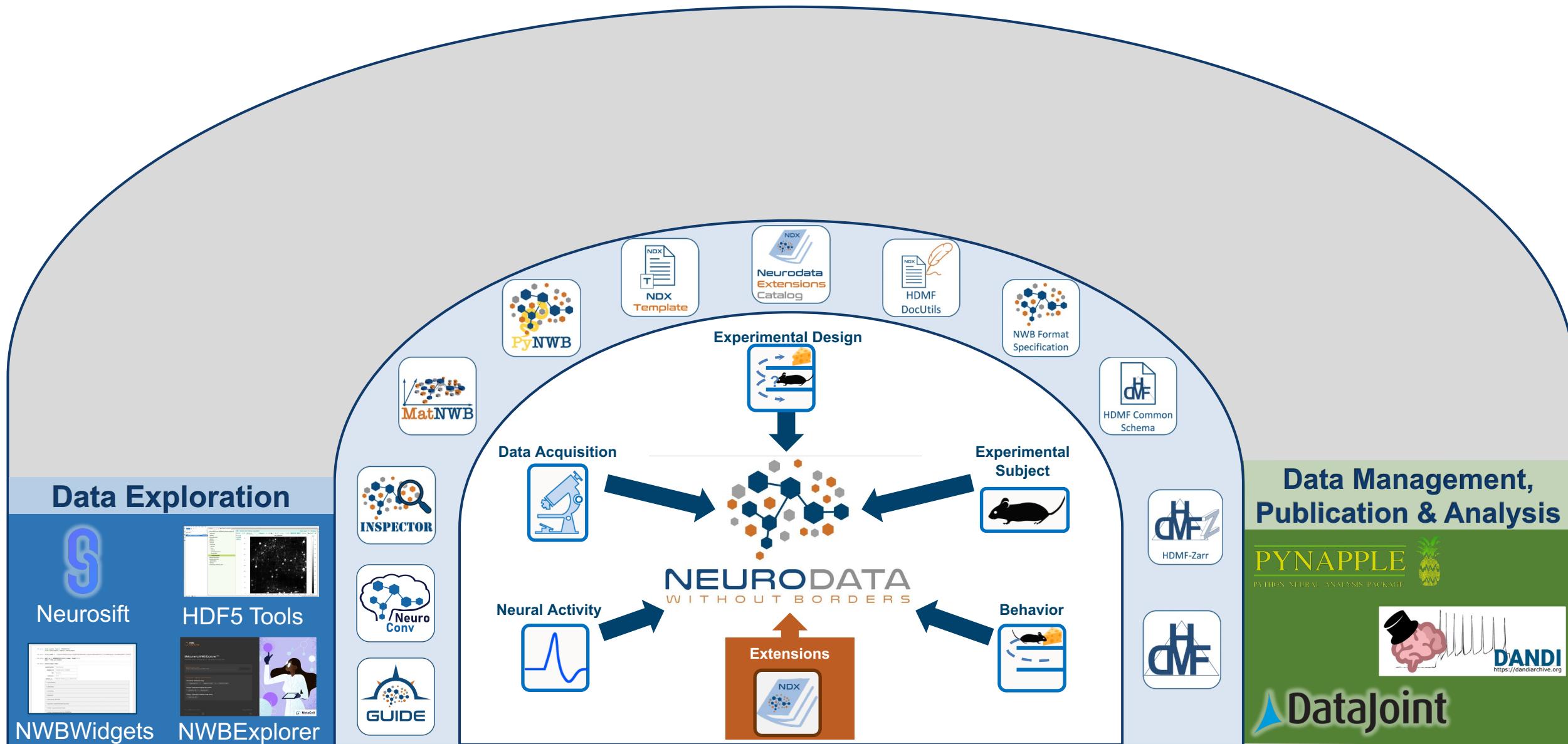
- Data standards are a critical conduit that facilitate:
 - Flow of data throughout the data lifecycle
 - Integration of data and software across phases of the data lifecycle
- NWB needs to support the needs of and integrate with technologies across the data lifecycle:
 - Work with (not compete with) existing and emerging data technologies
- NWB is a data standard for (not a standard of) neurophysiology experiments

NWB Data Analysis, Visualization, and Management Tools

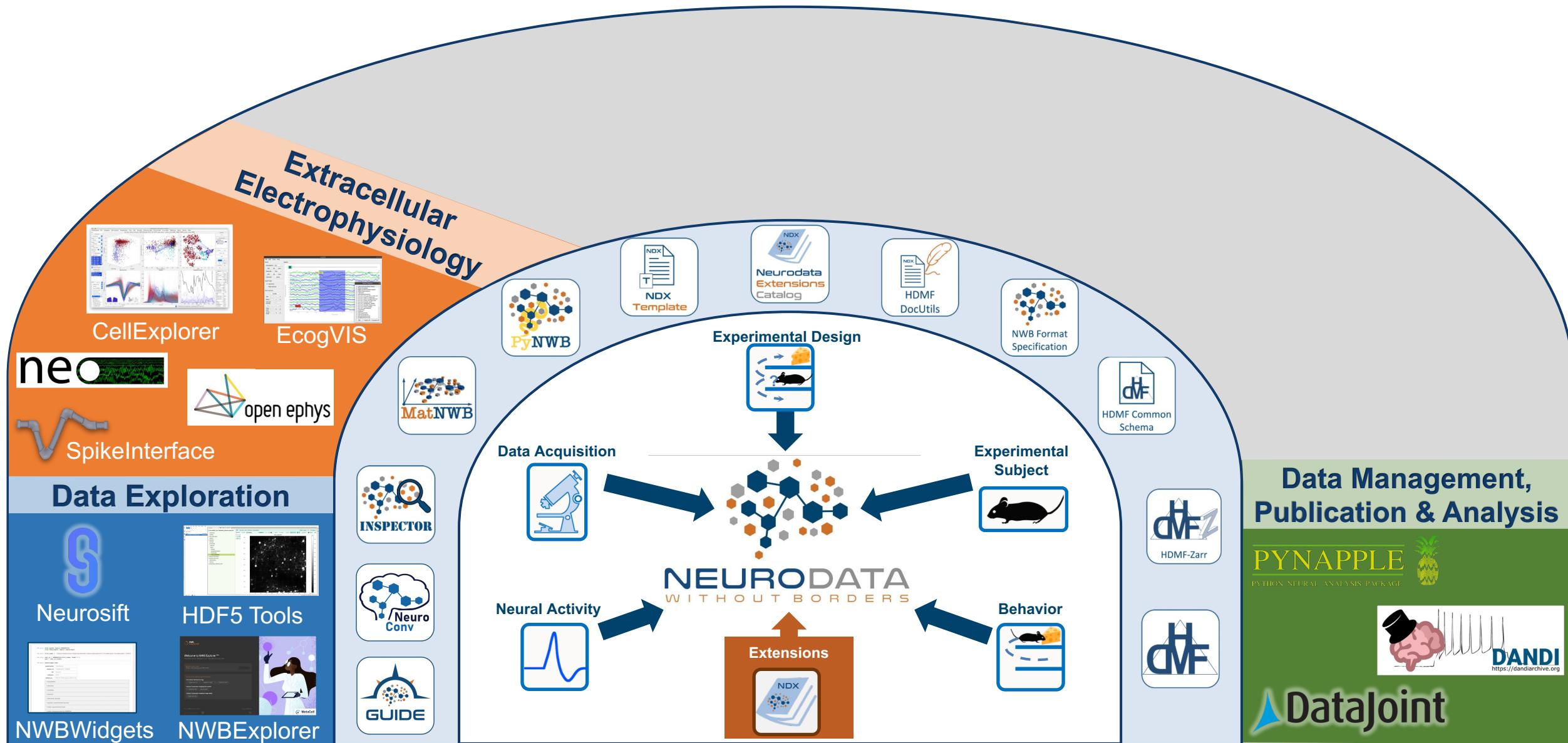
A unified data standard and software ecosystem for neurophysiology



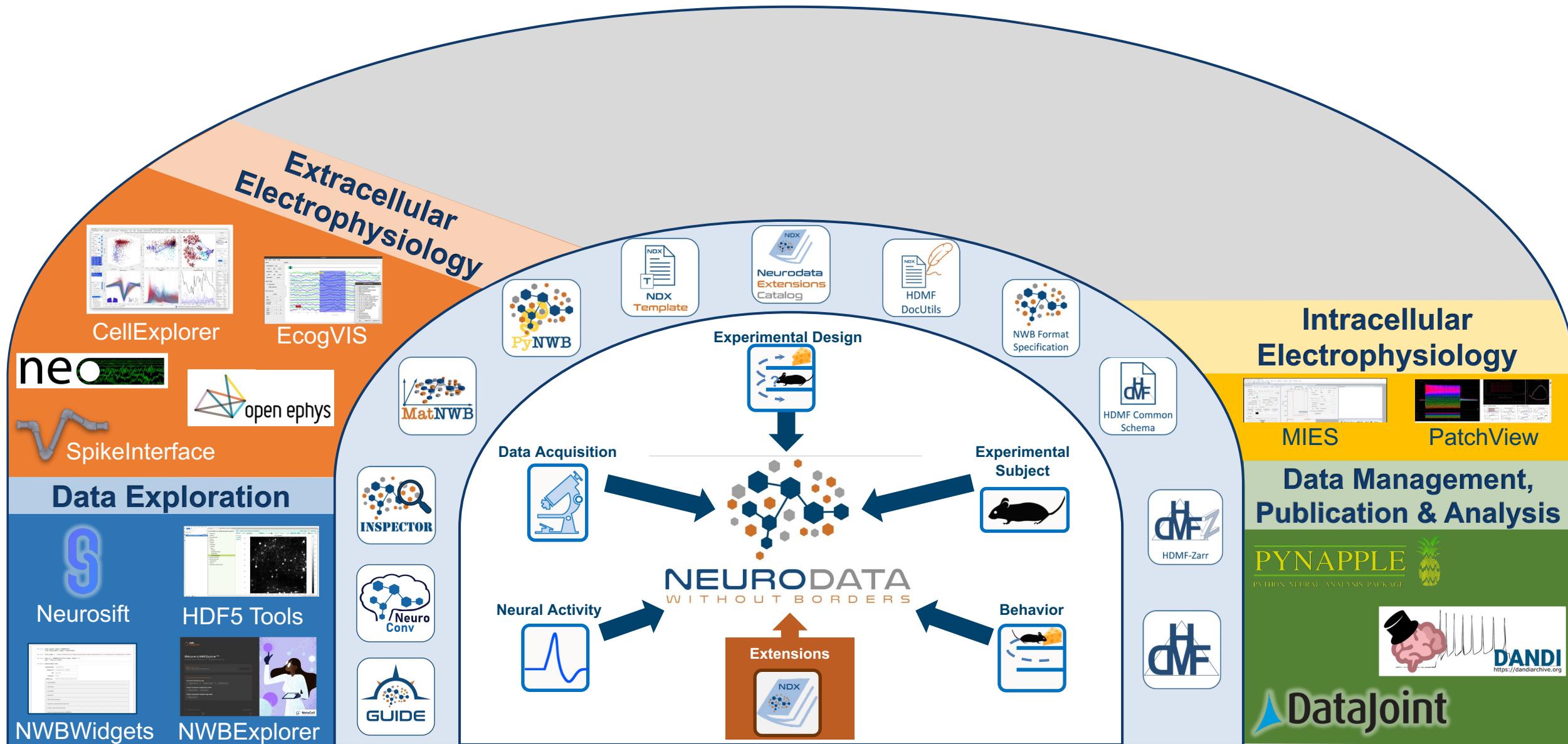
A unified data standard and software ecosystem for neurophysiology



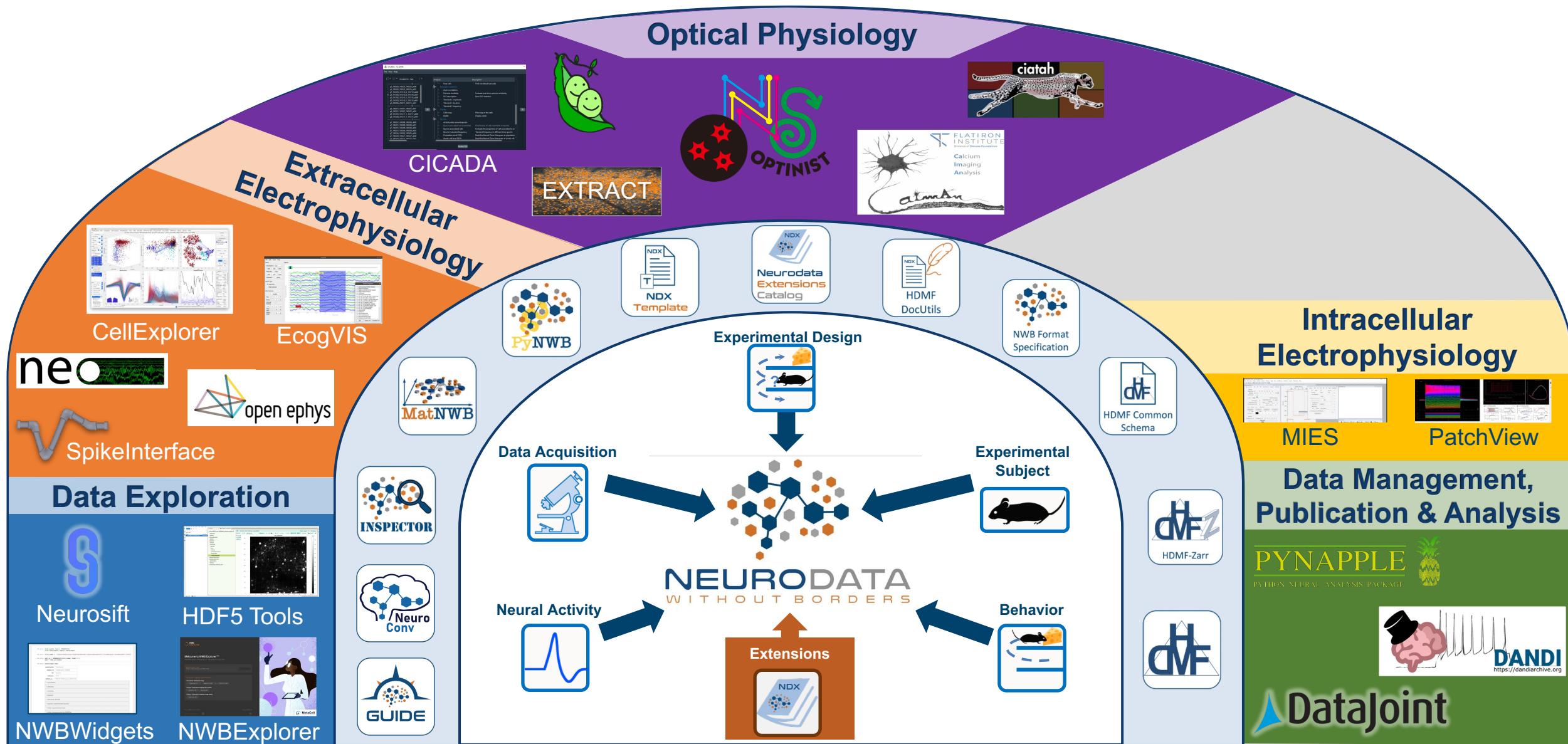
A unified data standard and software ecosystem for neurophysiology



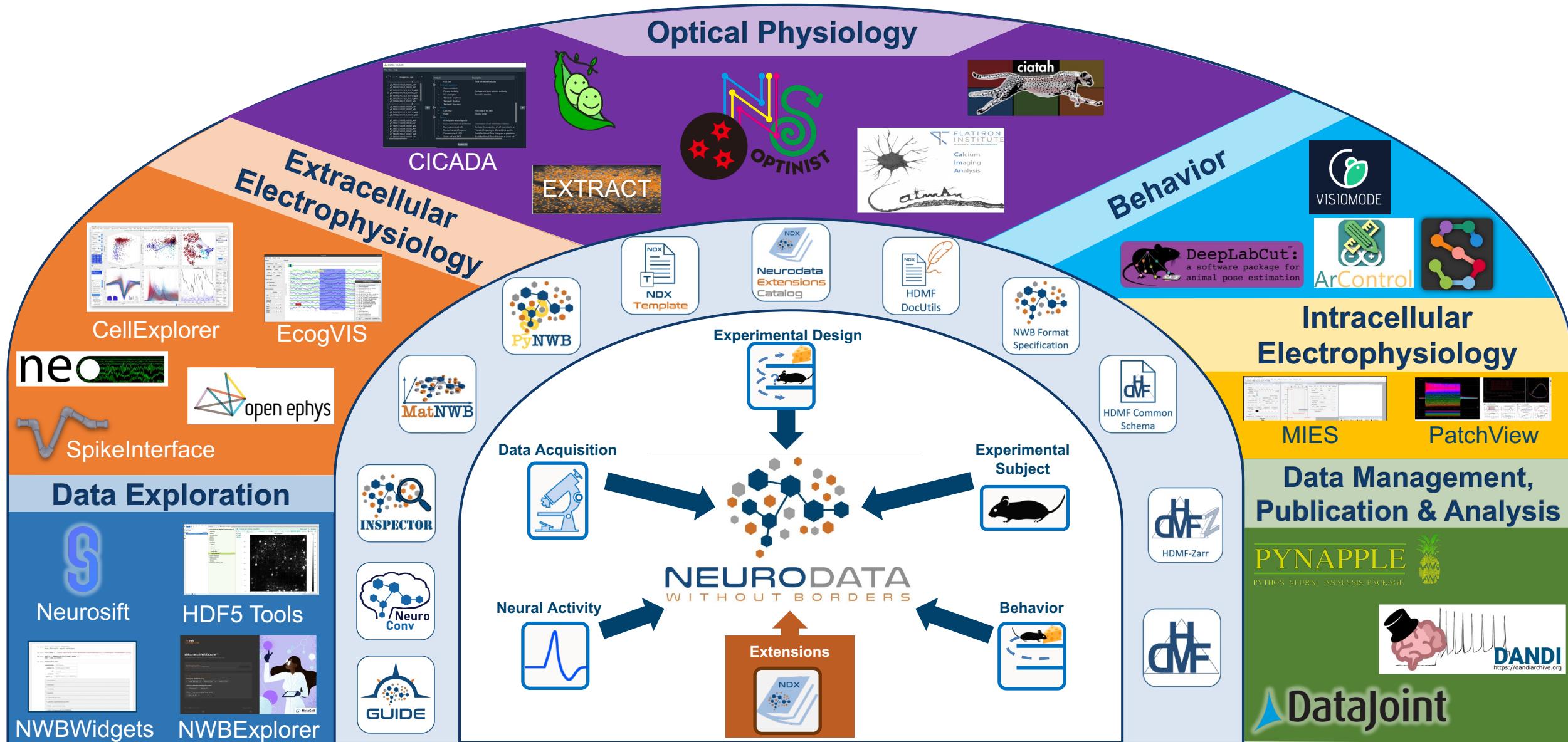
A unified data standard and software ecosystem for neurophysiology



A unified data standard and software ecosystem for neurophysiology



A unified data standard and software ecosystem for neurophysiology



Tools Overview: <https://nwb-overview.readthedocs.io>

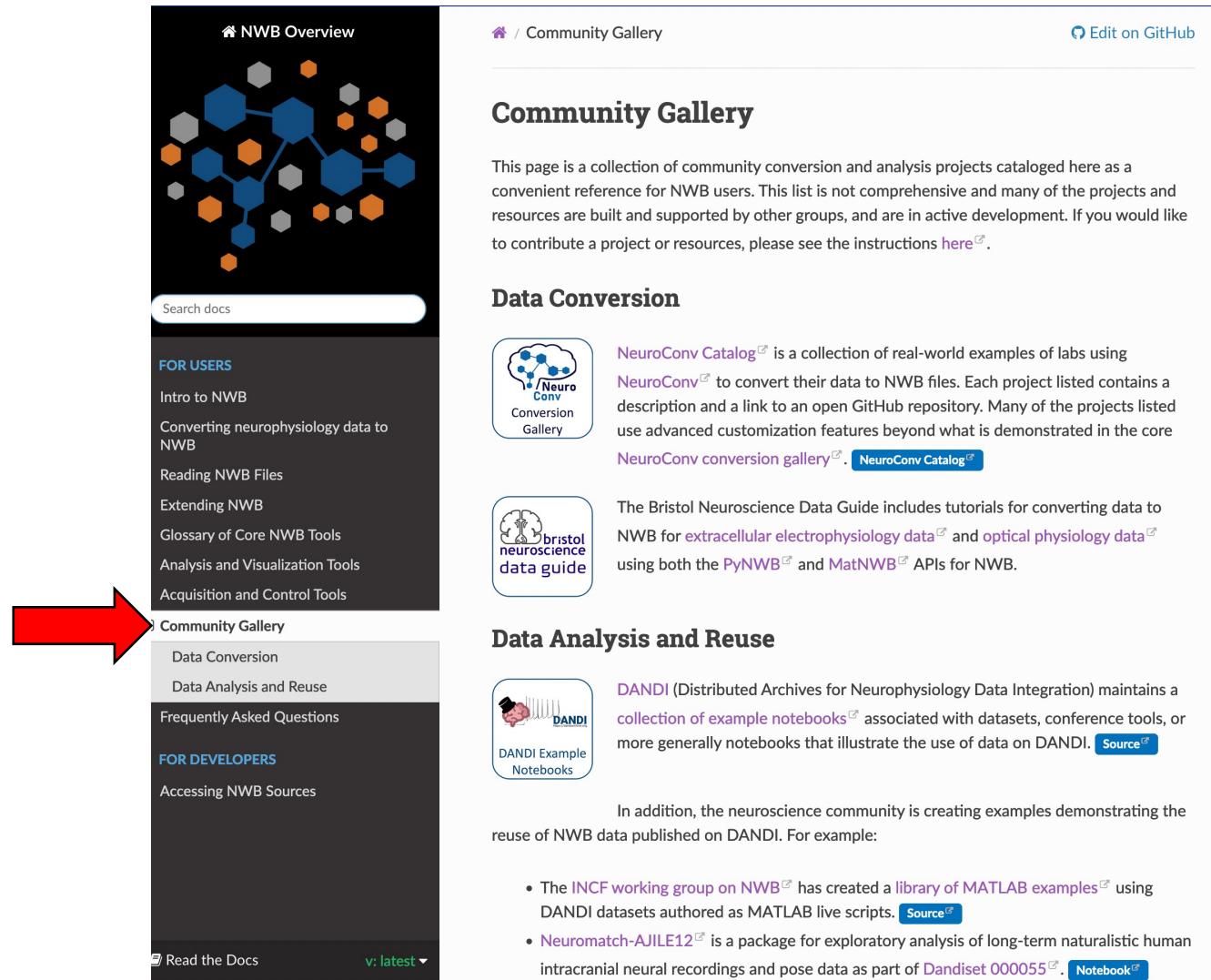
The screenshot shows the NWB Overview page with a dark header and sidebar. The sidebar contains sections for 'FOR USERS' (Intro to NWB, Converting neurophysiology data to NWB, Reading NWB Files, Extending NWB) and 'FOR DEVELOPERS' (Glossary of Core NWB Tools, Read/Write NWB File APIs, Converting Data to NWB, NeuroConv). A red arrow points from the 'FOR USERS' section towards the central content area.

This page provides a glossary of core NWB tools. It includes sections for 'Read/Write NWB File APIs' (PyNWB and MatNWB), 'Converting Data to NWB' (NeuroConv), and a general overview of the core NWB stack. It also links to the 'Analysis and Visualization Tools' page.

This page lists various analysis and visualization tools. A red arrow points from the 'Analysis and Visualization Tools' section in the sidebar of the previous page towards this page. The listed tools include Intro to NWB, Converting neurophysiology data to NWB, Reading NWB Files, Extending NWB, Glossary of Core NWB Tools, NWB Widgets, Neurosift, NWB Explorer, HDF Tools, SpikelInterface, CellExplorer, EcogVIS, Neo, CalmAn, suite2p, CIAtah, EXTRACT, CICADA, OptiNiSt, PatchView, DeepLabCut, SLEAP, pynapple, DANDI, Acquisition and Control Tools, Community Gallery, Frequently Asked Questions, and FOR DEVELOPERS (Accessing NWB Sources).

This page provides detailed information about specific tools. It includes sections for 'Exploring NWB Files' (NWB Widgets, Neurosift, NWB Explorer, HDF Tools), 'Analysis and Visualization Tools' (Introduction, PyNWB, MatNWB, NeuroConv), and 'FOR DEVELOPERS' (Accessing NWB Sources). It also features a 'Source' link for the instructions.

Community Gallery: <https://nwb-overview.readthedocs.io>



The screenshot shows the NWB Overview documentation page. On the left, there's a sidebar with navigation links for users and developers. A red arrow points from the sidebar to the "Community Gallery" section on the main content page. The main content page has a header "Community Gallery" and a sub-section "Data Conversion". It features a "NeuroConv Catalog" card with a brain icon and a "bristol neuroscience data guide" card with a brain icon. Below these are sections for "Data Analysis and Reuse" and "DANDI Example Notebooks". The footer includes "Read the Docs" and a "v: latest" dropdown.

NWB Overview

FOR USERS

- Intro to NWB
- Converting neurophysiology data to NWB
- Reading NWB Files
- Extending NWB
- Glossary of Core NWB Tools
- Analysis and Visualization Tools
- Acquisition and Control Tools

Community Gallery

- Data Conversion
- Data Analysis and Reuse
- Frequently Asked Questions

FOR DEVELOPERS

- Accessing NWB Sources

Read the Docs v: latest

Community Gallery

Edit on GitHub

Community Gallery

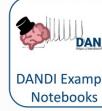
This page is a collection of community conversion and analysis projects cataloged here as a convenient reference for NWB users. This list is not comprehensive and many of the projects and resources are built and supported by other groups, and are in active development. If you would like to contribute a project or resources, please see the instructions [here](#).

Data Conversion

 NeuroConv Catalog is a collection of real-world examples of labs using NeuroConv to convert their data to NWB files. Each project listed contains a description and a link to an open GitHub repository. Many of the projects listed use advanced customization features beyond what is demonstrated in the core NeuroConv conversion gallery. [NeuroConv Catalog](#)

 The Bristol Neuroscience Data Guide includes tutorials for converting data to NWB for [extracellular electrophysiology data](#) and [optical physiology data](#) using both the [PyNWB](#) and [MatNWB](#) APIs for NWB.

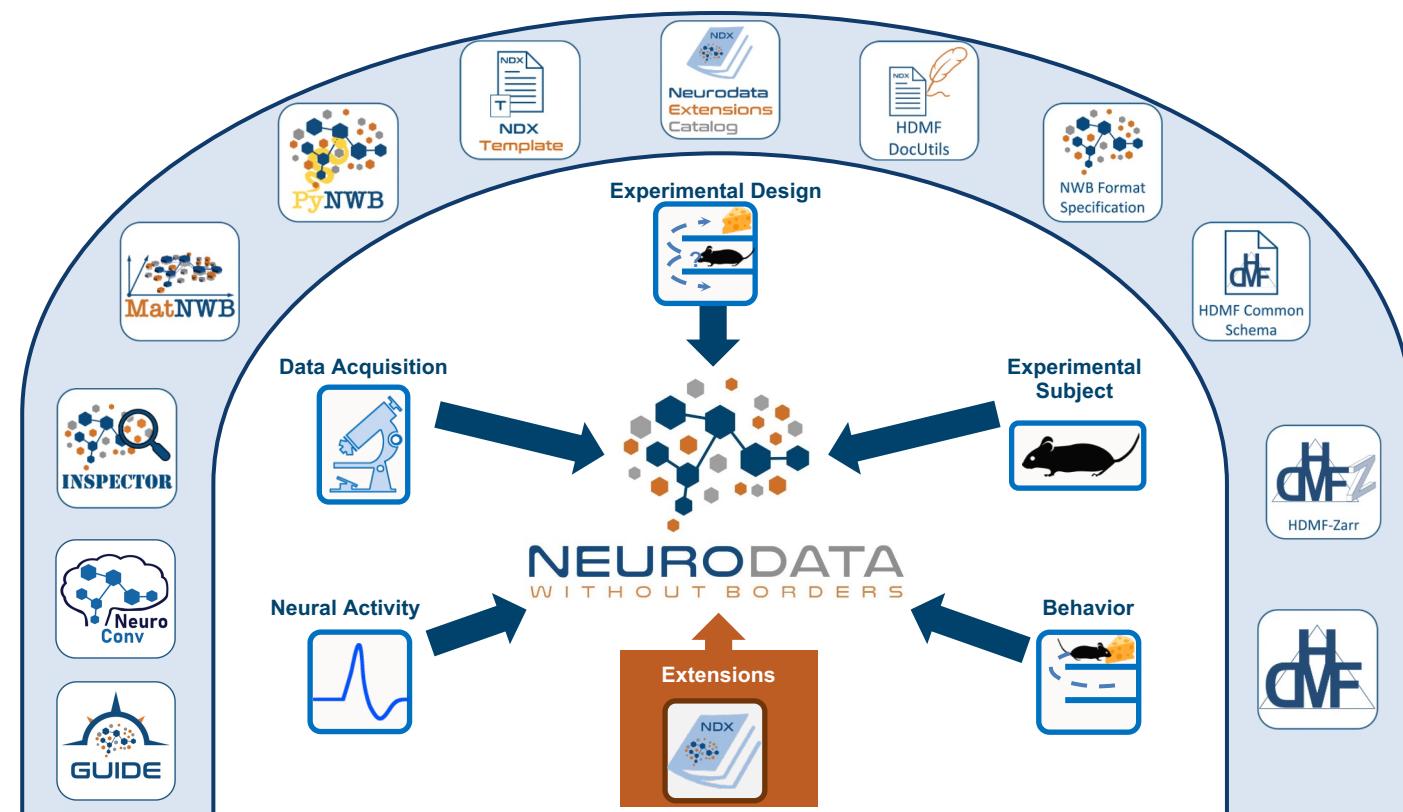
Data Analysis and Reuse

 DANDI (Distributed Archives for Neurophysiology Data Integration) maintains a collection of example notebooks associated with datasets, conference tools, or more generally notebooks that illustrate the use of data on DANDI. [Source](#)

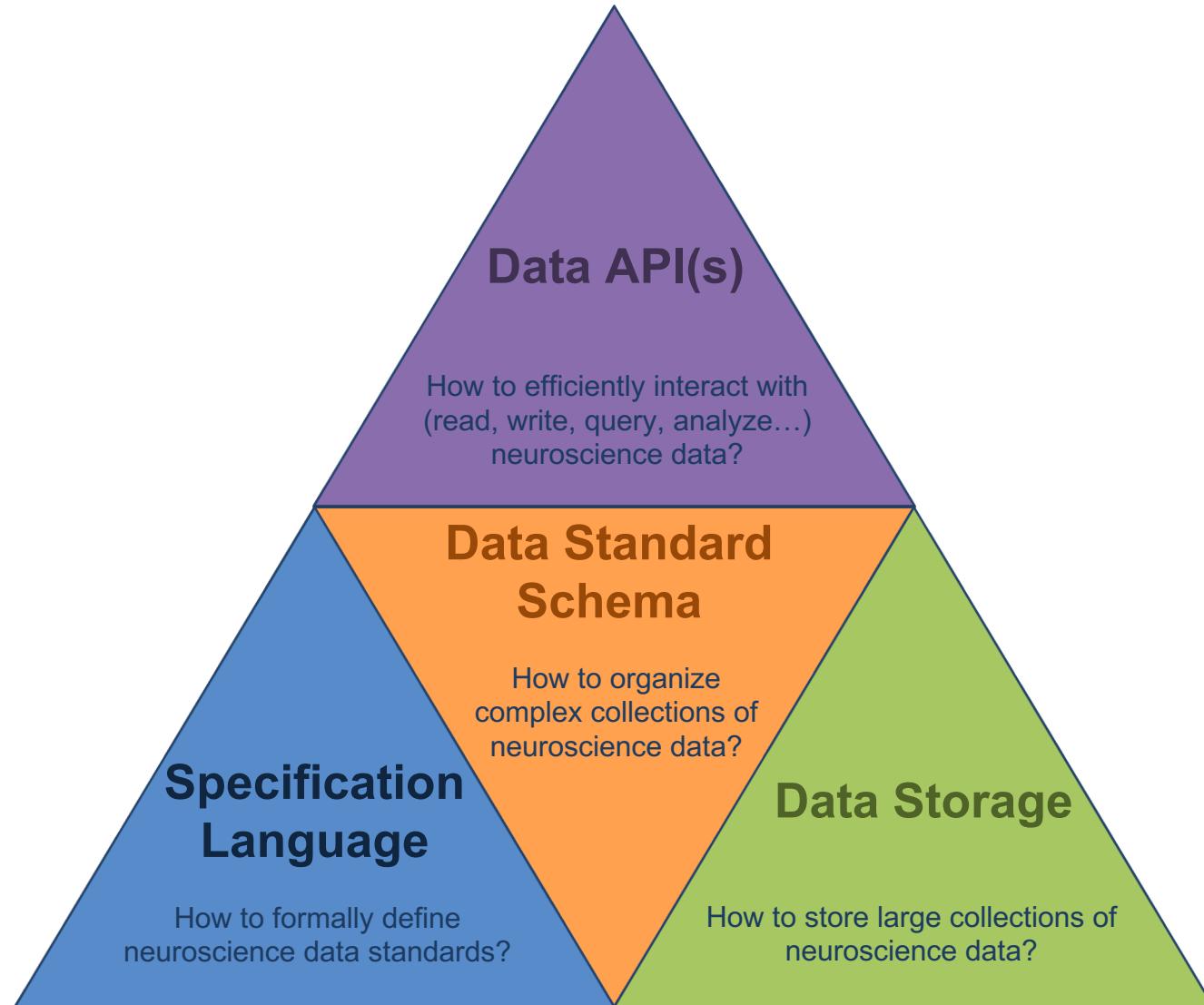
In addition, the neuroscience community is creating examples demonstrating the reuse of NWB data published on DANDI. For example:

- The INCF working group on NWB has created a library of MATLAB examples using DANDI datasets authored as MATLAB live scripts. [Source](#)
- Neuromatch-AJILE12 is a package for exploratory analysis of long-term naturalistic human intracranial neural recordings and pose data as part of Dandiset 000055. [Notebook](#)

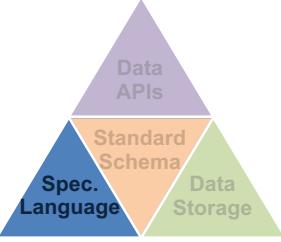
Core NWB Data Standard Ecosystem



Main components of the NWB ecosystem

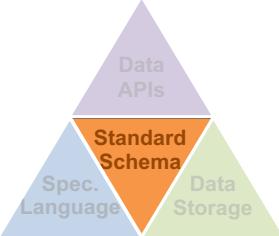


Specification Language



- Defines data objects (Group, Dataset, Link, Attribute) and their features (name, data type, dimensions, description, etc.) used to model data
- Defines common data elements as `neurodata_types` (object-oriented)
 - Supports inheritance and composition of types
- Extensions to NWB are defined using same language and formal rule





NWB Standard Schema

- Uses the [NWB Specification Language](#) to define a large collection of reusable `neurodata_types` for storing a large set of neurophysiology data types and to define their organization as part of the NWBFile hierarchy



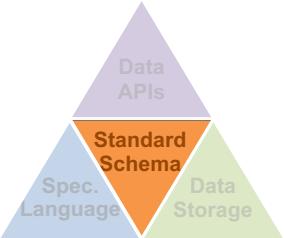
```

L general
  L subject
    neurodata_type: Subject
    L species : Mouse
    L experimenter :
      • Oliver Rübel
      • Ryan Ly
    L extracellular_ephs
      L electrodes : 

| x   | y   | z   | imp | location  | ... |
|-----|-----|-----|-----|-----------|-----|
| 0.1 | 0.5 | 0.1 | ..  | SSp-II1   | ... |
| 1.1 | 0.5 | 2.1 | ..  | SSp-II1   | ... |
| 0.5 | 1.0 | 1.1 | ..  | SSp-II2/3 | ... |

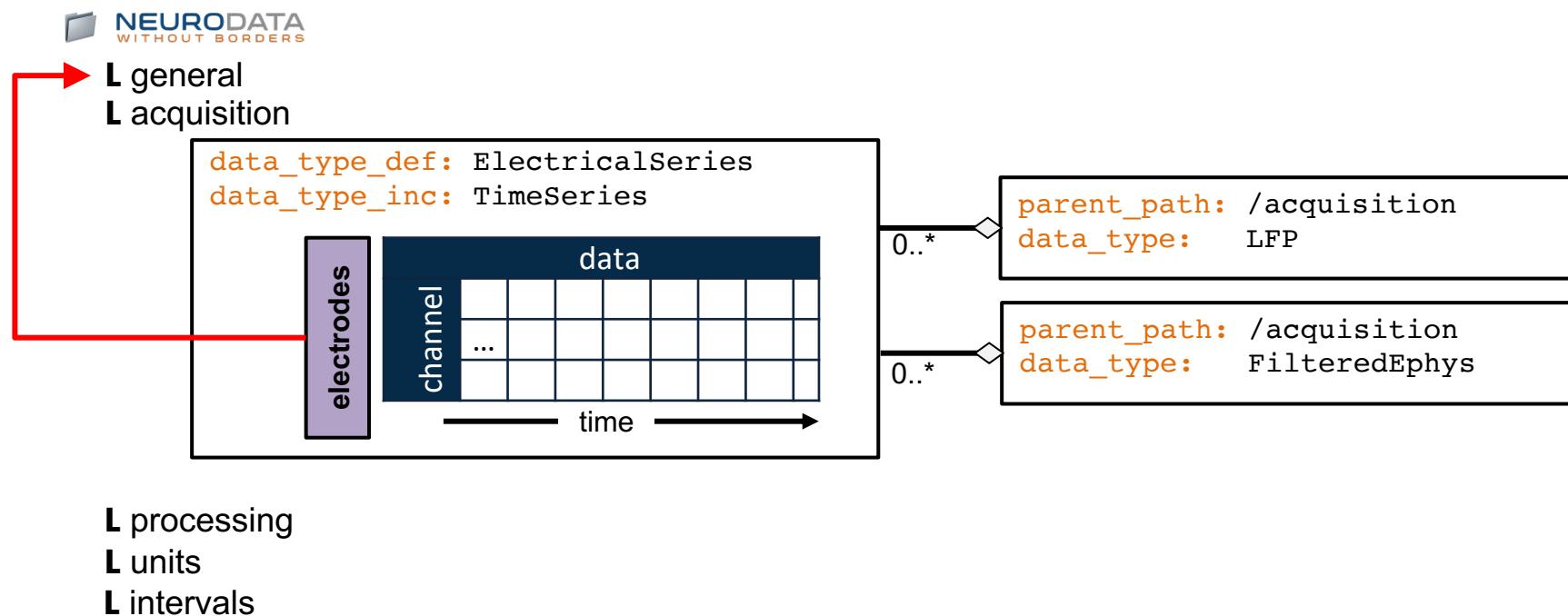

      L acquisition
      L processing
      L units
      L intervals
      ...
  
```

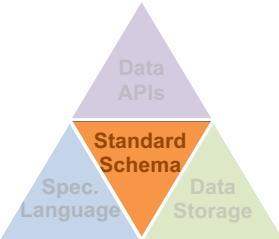




NWB Standard Schema

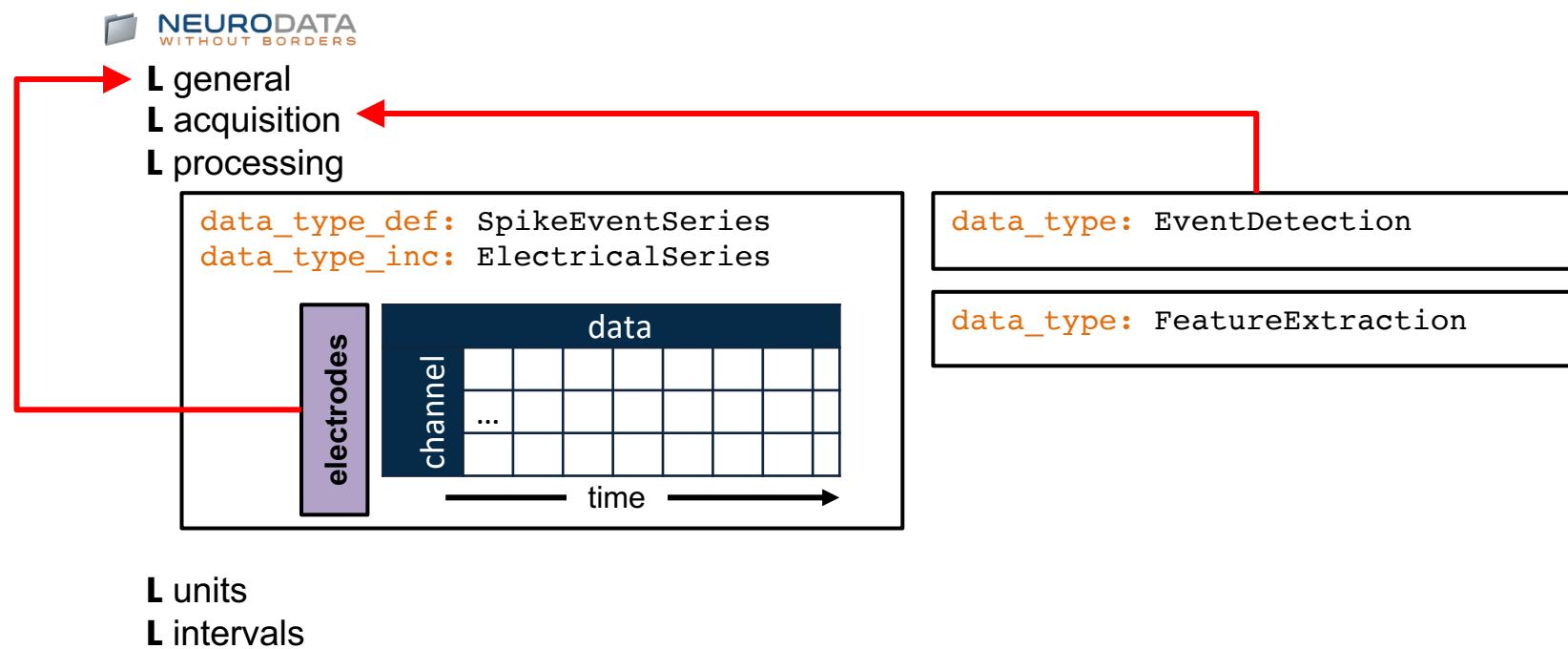
- Uses the [NWB Specification Language](#) to define a large collection of reusable `neurodata_types` for storing a large set of neurophysiology data types and to define their organization as part of the NWBFile hierarchy

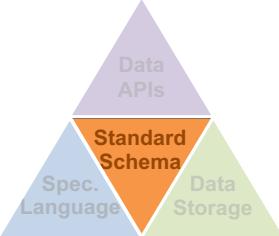




NWB Standard Schema

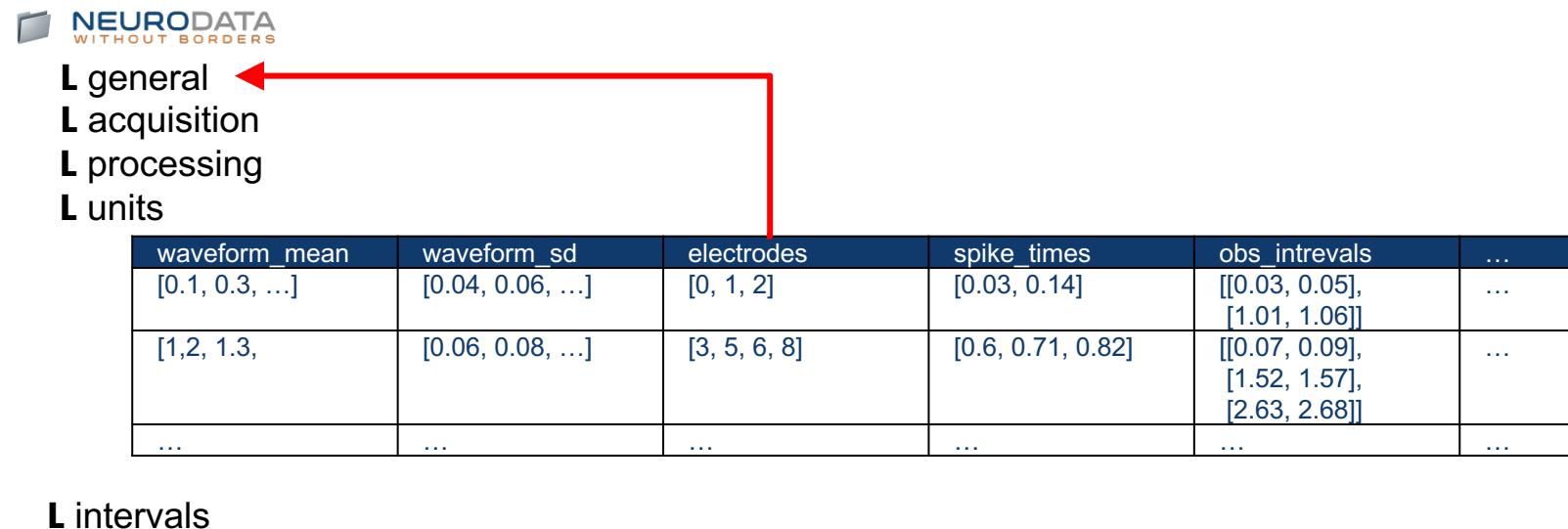
- Uses the [NWB Specification Language](#) to define a large collection of reusable `neurodata_types` for storing a large set of neurophysiology data types and to define their organization as part of the NWBFile hierarchy



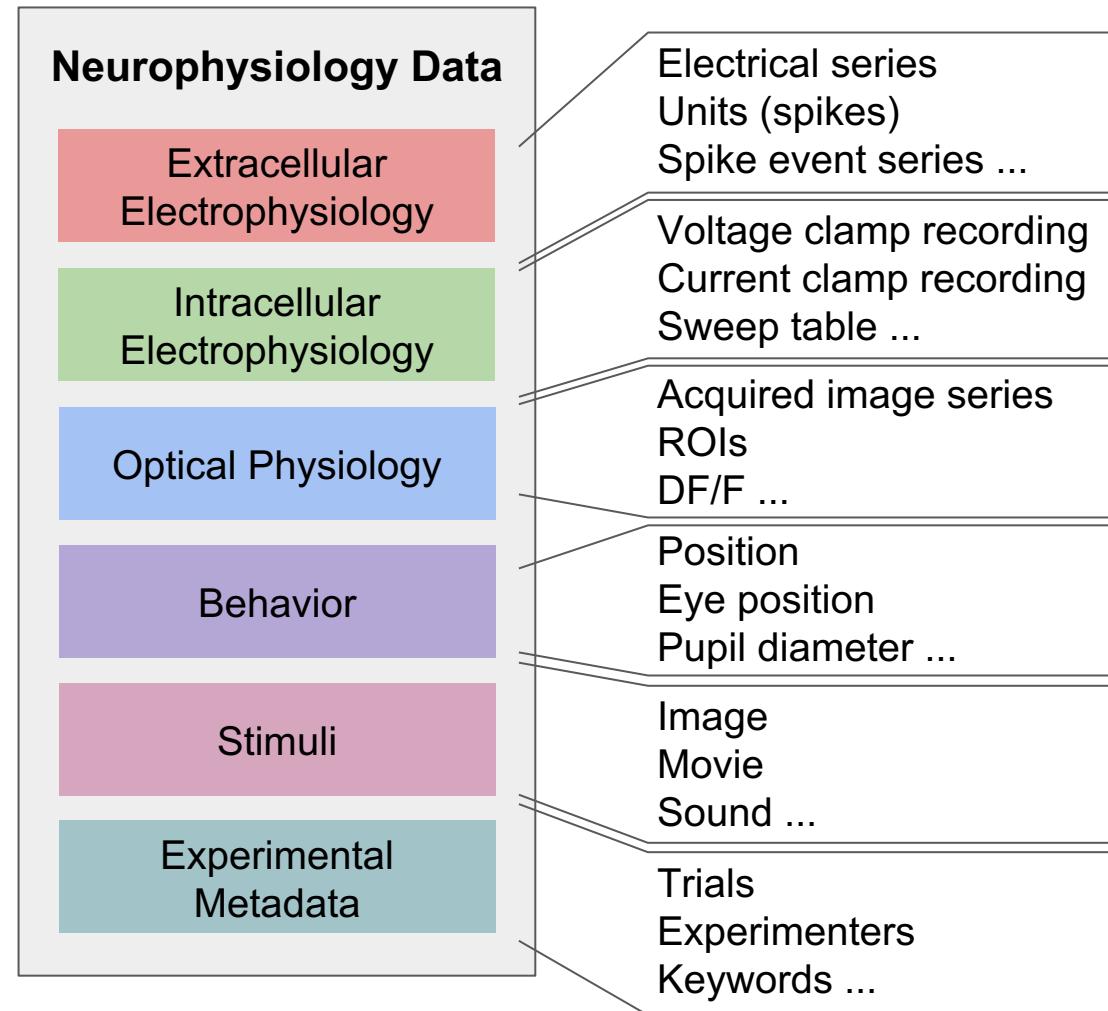
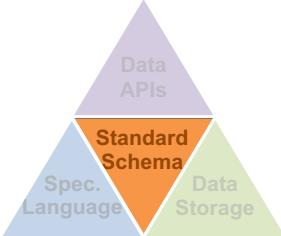


NWB Standard Schema

- Uses the [NWB Specification Language](#) to define a large collection of reusable `neurodata_types` for storing a large set of neurophysiology data types and to define their organization as part of the NWBFile hierarchy

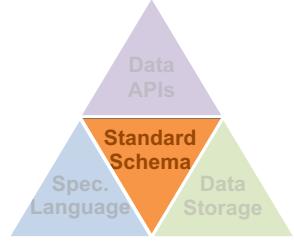


NWB provides structured `neurodata_types` for most of neurophysiology data



NWB Extensions (NDX)

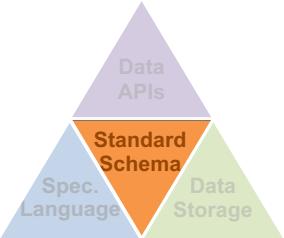
- Using the [NWB Specification Language](#), NWB supports the formal extension of the data standard to define new data types and metadata



Creating the extension

```
from pynwb.spec import NWBDatasetSpec, NWBNamespaceBuilder, NWBGroupSpec, NWBAttributeSpec
surface = NWBGroupSpec(neurodata_type_def='Surface',
                       neurodata_type_inc='NWBDataInterface',
                       quantity='+', doc='brain cortical surface')
surface.add_dataset(NWBDatasetSpec(
    doc='...', shape=(None, 3),
    name='faces', dtype='uint', dims=...))
surface.add_dataset(NWBDatasetSpec(
    doc='...', shape=(None, 3),
    name='vertices', dtype='float', dims=...))
surface.add_attribute(...)

ns_builder = NWBNamespaceBuilder(doc=..., name='ecog',
                                 version='1.0', author='Ben Dichter', ...)
ns_builder.add_spec('ecog.extensions.yaml', surface)
ns_builder.export('ecog.namespace.yaml')
```



NWB Extensions (NDX)

- Using the [NWB Specification Language](#), NWB supports the formal extension of the data standard to define new data types and metadata

Creating the extension

```

from pynwb.spec import NWBDatasetSpec, NWBNamespaceBuilder,
    NWBGroupSpec, NWBAttributeSpec
surface = NWBGroupSpec(neurodata_type_def='Surface',
    neurodata_type_inc='NWBDatatypesInterface',
    quantity='+', doc='brain cortical surface')
surface.add_dataset(NWBDatasetSpec(
    doc='...', shape=(None, 3),
    name='faces', dtype='uint', dims=...))
surface.add_dataset(NWBDatasetSpec(
    doc='...', shape=(None, 3),
    name='vertices', dtype='float', dims=...))
surface.add_attribute(...)

ns_builder = NWBNamespaceBuilder(doc=..., name='ecog',
    version='1.0', author='Ben Dichter', ...)
ns_builder.add_spec('ecog.extensions.yaml', surface)
ns_builder.export('ecog.namespace.yaml')

```

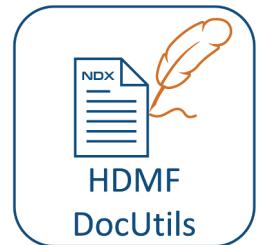
Using the extension

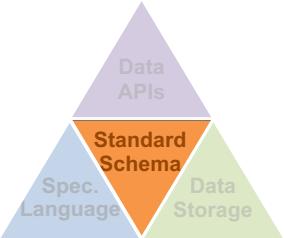
PyNWB: write

```

from pynwb import load_namespaces, get_class,
    NWBHDF5IO, NWBFile ...
nwbfile = NWBFile(...)
load_namespaces('ecog.namespace.yaml')
Surface = get_class('Surface', 'ecog')
surf = Surface(faces=..., vertices=...,
    name='Surface_1...')
nwbfile.add_acquisition(surf)
with NWBHDF5IO('surface_example.nwb', 'w') as io:
    io.write(nwbfile)

```

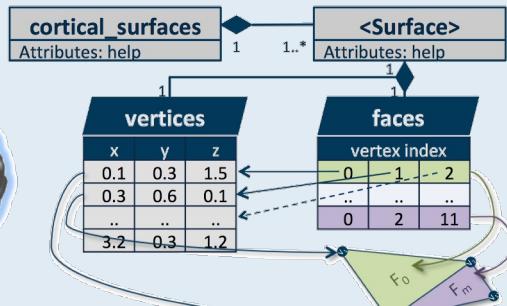
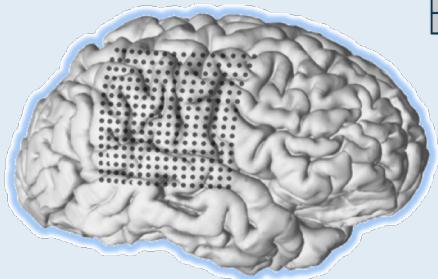




NWB Extensions (NDX)

- Using the [NWB Specification Language](#), NWB supports the formal extension of the data standard to define new data types and metadata

Creating the extension



```
from pynwb.spec import NWBDatasetSpec, NWBNamespaceBuilder,
                     NWBGroupSpec, NWBAttributeSpec
surface = NWBGroupSpec(neurodata_type_def='Surface',
                       neurodata_type_inc='NWBDatatypesInterface',
                       quantity='+', doc='brain cortical surface')
surface.add_dataset(NWBDatasetSpec(
    doc='...', shape=(None, 3),
    name='faces', dtype='uint', dims=...))
surface.add_dataset(NWBDatasetSpec(
    doc='...', shape=(None, 3),
    name='vertices', dtype='float', dims=...))
surface.add_attribute(...)

ns_builder = NWBNamespaceBuilder(doc=..., name='ecog',
                                 version='1.0', author='Ben Dichter', ...)
ns_builder.add_spec('ecog.extensions.yaml', surface)
ns_builder.export('ecog.namespace.yaml')
```

Using the extension

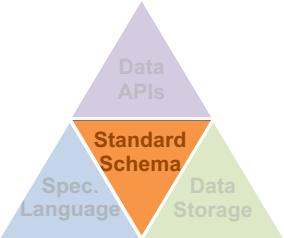
PyNWB: write

```
from pynwb import load_namespaces, get_class,
                 NWBHDF5IO, NWBFile ...
nwbfile = NWBFile(...)
load_namespaces('ecog.namespace.yaml')
Surface = get_class('Surface', 'ecog')
surf = Surface(faces=..., vertices=...,
               name='Surface_1')
nwbfile.add_acquisition(surf)
with NWBHDF5IO('surface_example.nwb', 'w') as io:
    io.write(nwbfile)
```

PyNWB: read

```
load_namespaces('ecog.namespace.yaml')
io = NWBHDF5IO('surface_example.nwb', 'r')
nwbfile = io.read()
nwbfile.get_acquisition('Surface 1').vertices
```

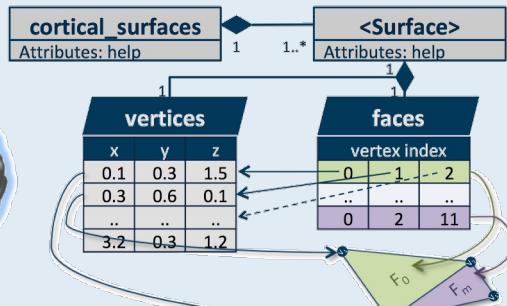
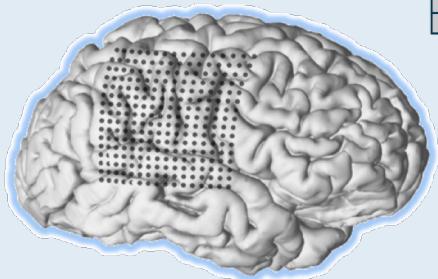




NWB Extensions (NDX)

- Using the [NWB Specification Language](#), NWB supports the formal extension of the data standard to define new data types and metadata

Creating the extension



```

from pynwb.spec import NWBDatasetSpec, NWBNamespaceBuilder,
                     NWBGroupSpec, NWBAttributeSpec
surface = NWBGroupSpec(neurodata_type_def='Surface',
                       neurodata_type_inc='NWBDataInterface',
                       quantity='+', doc='brain cortical surface')
surface.add_dataset(NWBDatasetSpec(
    doc='...', shape=(None, 3),
    name='faces', dtype='uint', dims=...))
surface.add_dataset(NWBDatasetSpec(
    doc='...', shape=(None, 3),
    name='vertices', dtype='float', dims=...))
surface.add_attribute(...)

ns_builder = NWBNamespaceBuilder(doc=..., name='ecog',
                                 version='1.0', author='Ben Dichter', ...)
ns_builder.add_spec('ecog.extensions.yaml', surface)
ns_builder.export('ecog.namespace.yaml')
  
```

Using the extension

PyNWB: write

```

from pynwb import load_namespaces, get_class,
                 NWBHDF5IO, NWBFile ...
nwbfile = NWBFile(...)
load_namespaces('ecog.namespace.yaml')
Surface = get_class('Surface', 'ecog')
surf = Surface(faces=..., vertices=...,
               name='Surface_1')
nwbfile.add_acquisition(surf)
with NWBHDF5IO('surface_example.nwb', 'w') as io:
    io.write(nwbfile)
  
```

PyNWB: read

```

load_namespaces('ecog.namespace.yaml')
io = NWBHDF5IO('surface_example.nwb', 'r')
nwbfile = io.read()
nwbfile.get_acquisition('Surface 1').vertices
  
```

MatNWB: write

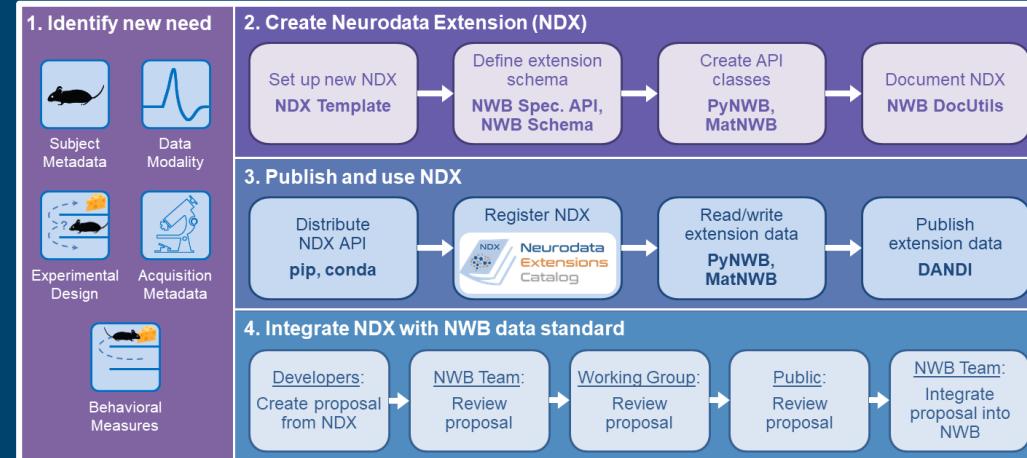
```

generateExtension('ecog.namespace.yaml');
file = nwbfile(...)
surf = types.ecog.Surface(
    'faces', faces, 'vertices',
    vertices);
file.acquisition.set('Surfaces_1', surf);
nwbExport(file, 'ecephys_tutorial.nwb')
  
```

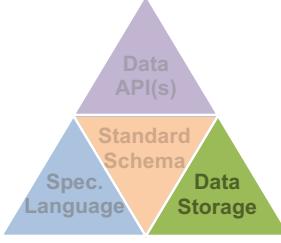


Neurodata Extension Catalog

- Provide easy-to-use tools and processes for creating, using, sharing, evaluation, reviewing, and extensions to the NWB data standard:
 - **Extension API:** Using the NWB Specification Language and API, NWB supports the formal extension of the data standard to define new data types and metadata
 - **NDX Catalog:** Online catalog for sharing and reviewing extensions:
 - **ndx_template:** Cookiecutter-based template for creating new extension code repositories, schema, documentation, and interfaces
 - **staged_extensions:** Submit extensions to the NDX catalog
 - **hdfm_docutils:** Tools to auto-generate documentation from extension schema
 - **Guidelines and rules:** for versioning, sharing, proposing, and reviewing extension
- Using the concept of Neurodata Extensions (NDX) enables:
 - users to adopt NWB and share their data even if they need to store additional metadata or new data types
 - Curation of the NWB data standard via community proposals
 - Complete testing and evaluation of proposals prior to integration with the NWB core standard and software API



NWB Data Storage

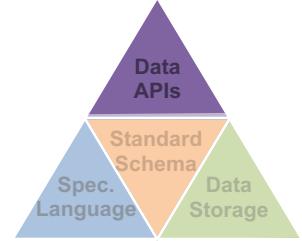


The data storage maps [NWB Specification Language](#) primitives (Groups, Datasets, Attributes etc.) to storage

- NWB uses HDF5 as its main file storage backend:
 - Supports large-scale storage of complex data collections in a single file
 - Optimized for performance (parallel I/O, advanced I/O filters etc.)
 - Supported across platforms and languages (Matlab, Python, C/C++, Fortran, R...)
 - Targets long-term support
 - Supports advanced I/O features (e.g., lazy load, chunking, compression, links and references etc.)
- Working with the community to explore and support integration with other data management and storage technologies:
 - HDMF-Zarr
 - DataJoint



Data APIs



PyNWB (Python) and MatNWB (Matlab) reference APIs:

- Efficiently read, write, query, and analyze NWB files
- APIs are interoperable
- APIs support advanced I/O features, e.g., lazy data load, chunking, compression, iterative data write, streaming etc.

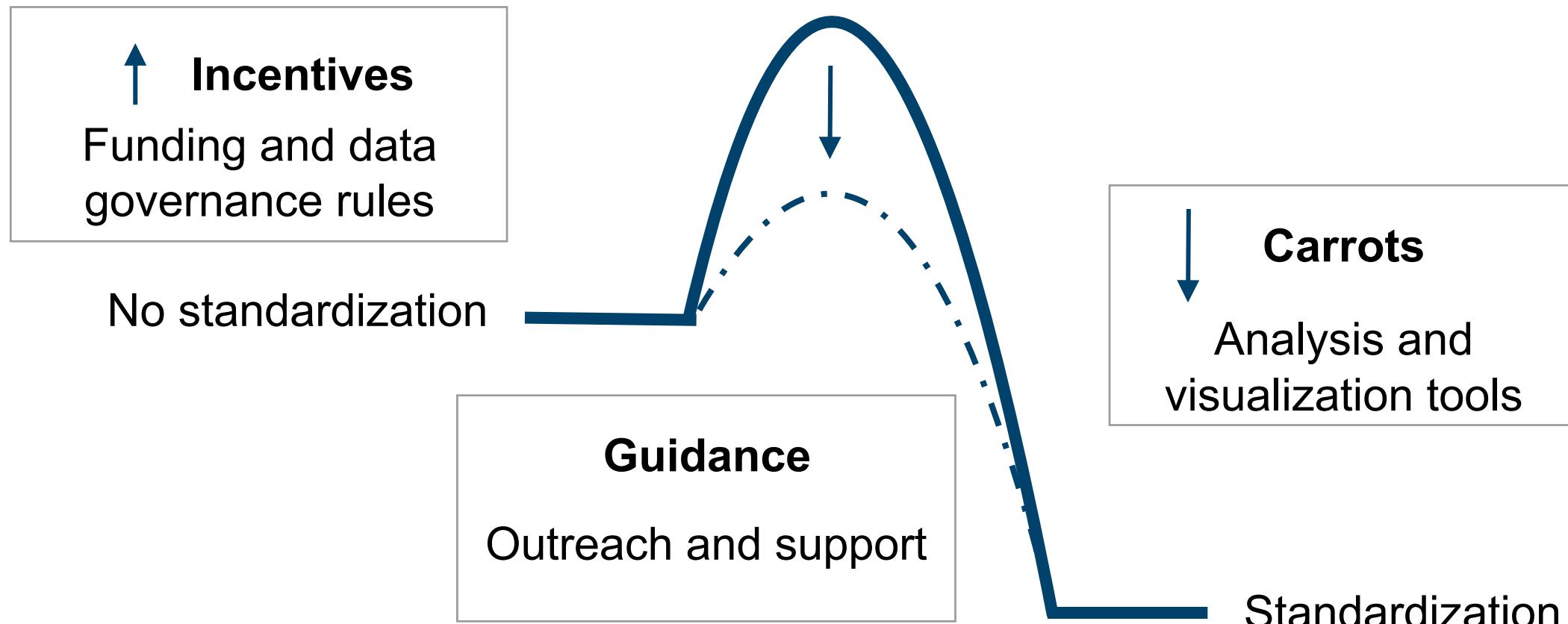


Convert data to NWB with support for many common acquisition and processing data formats



Validate and inspect NWB files

Overcoming the energy barrier in data standardization to advance FAIR neurophysiology data through dissemination of NWB



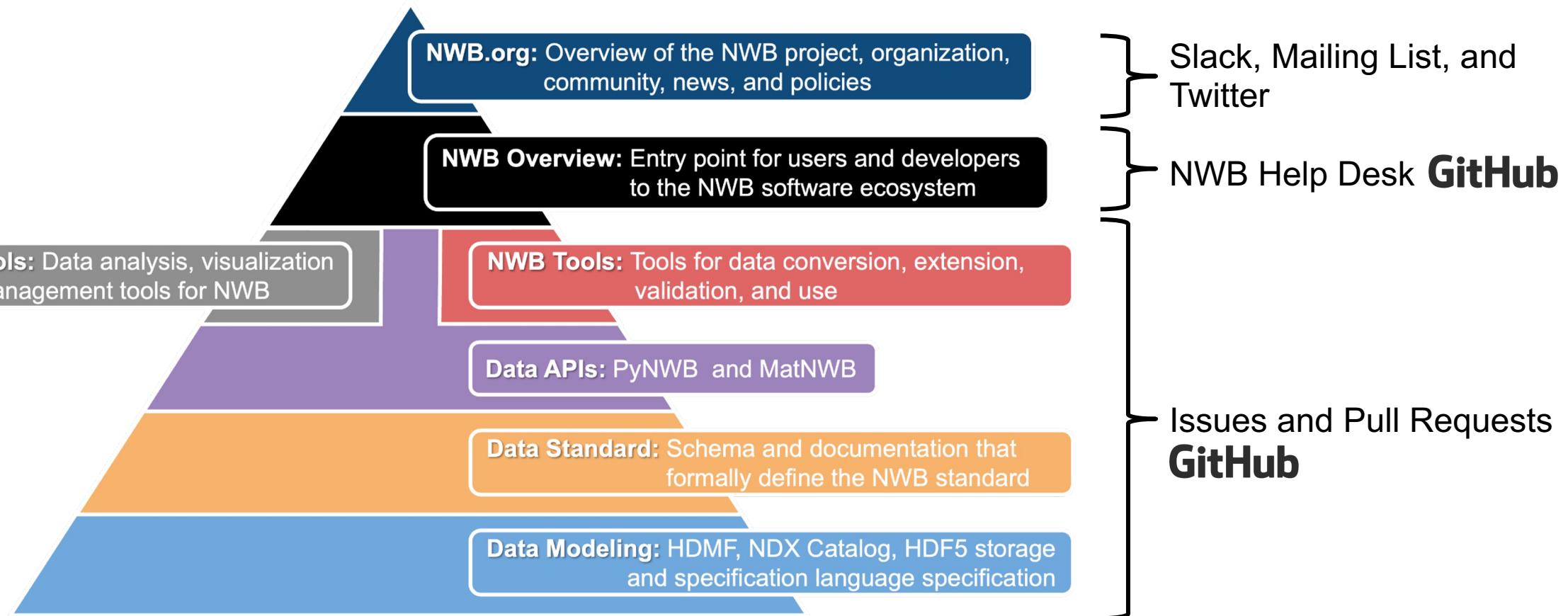
Enable neuroscience researchers to effectively utilize the NWB data standard and software

Goal: Lower the barriers of adopting NWB and promote broad dissemination of NWB by focusing on the needs of neuroscience researchers and labs by:

- Online and in-person training
 - July 24 – 26: [NWB User Days](#) @ HHMI Janelia (Ashburn, VA, USA)
 - July 27 – 29: [NWB Developer Days](#) @ HHMI Janelia (Ashburn, VA, USA)
 - Sept. 5 – 8: [NeuroDataReHack](#) @ IRBO World Conference (Granada, Spain)
- Consulting and support for labs to integrate with NWB
- Create community-driven processes for integration of new neuroscience technologies with NWB
 - Founded NWB Technical Advisory Board (TAB)
 - Growing collection of community extensions, e.g., ndx-pose, ndx-beadl, ndx-events, ndx-geneotype, etc.

Navigating the online documentation

How to get help?



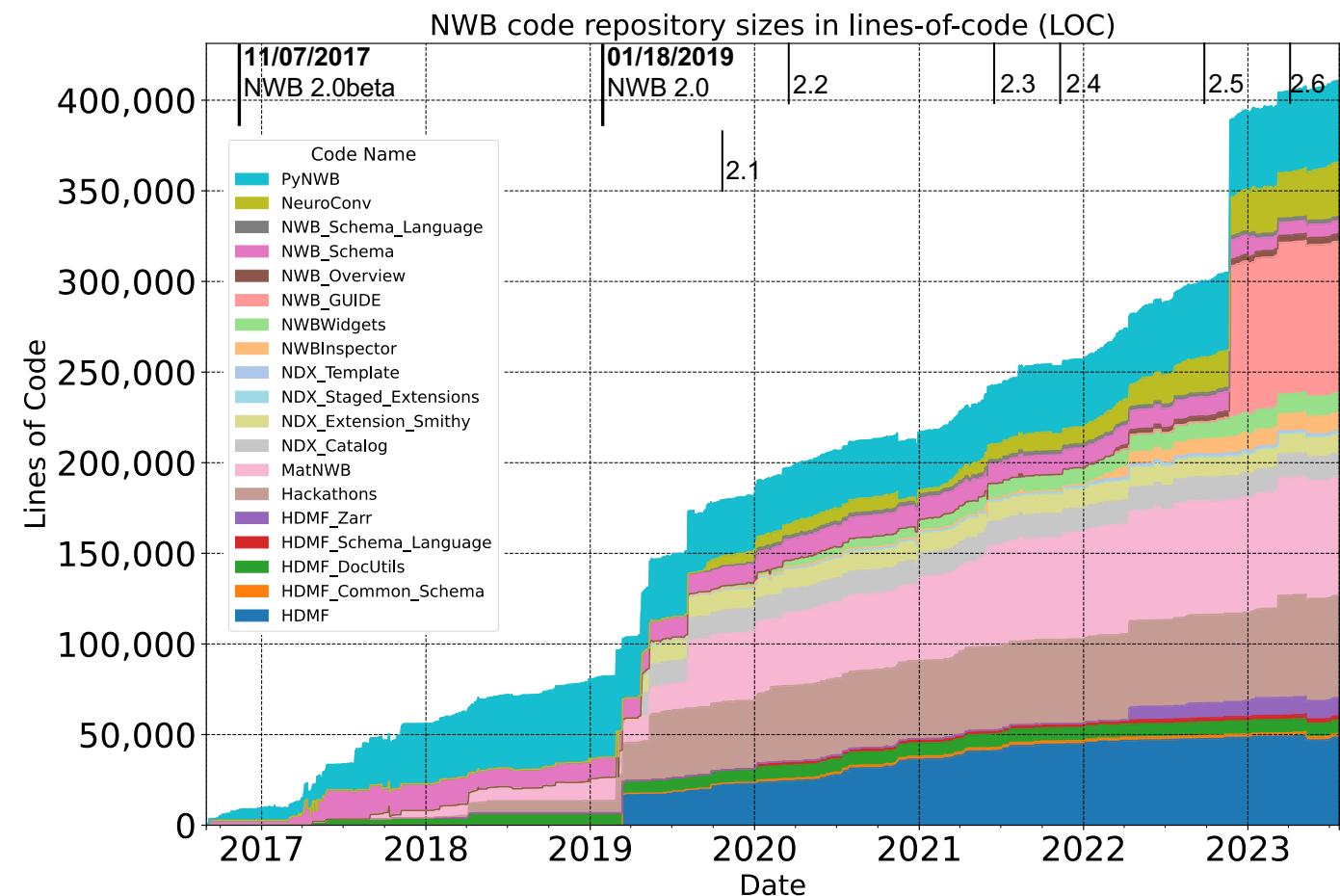
Enable integration of NWB with neurophysiology data analysis and management tools and technologies

Goal: Promote and support integration of NWB with neuroscience tools creating a vibrant software ecosystem around NWB and creating incentive for researchers to adopt NWB through:

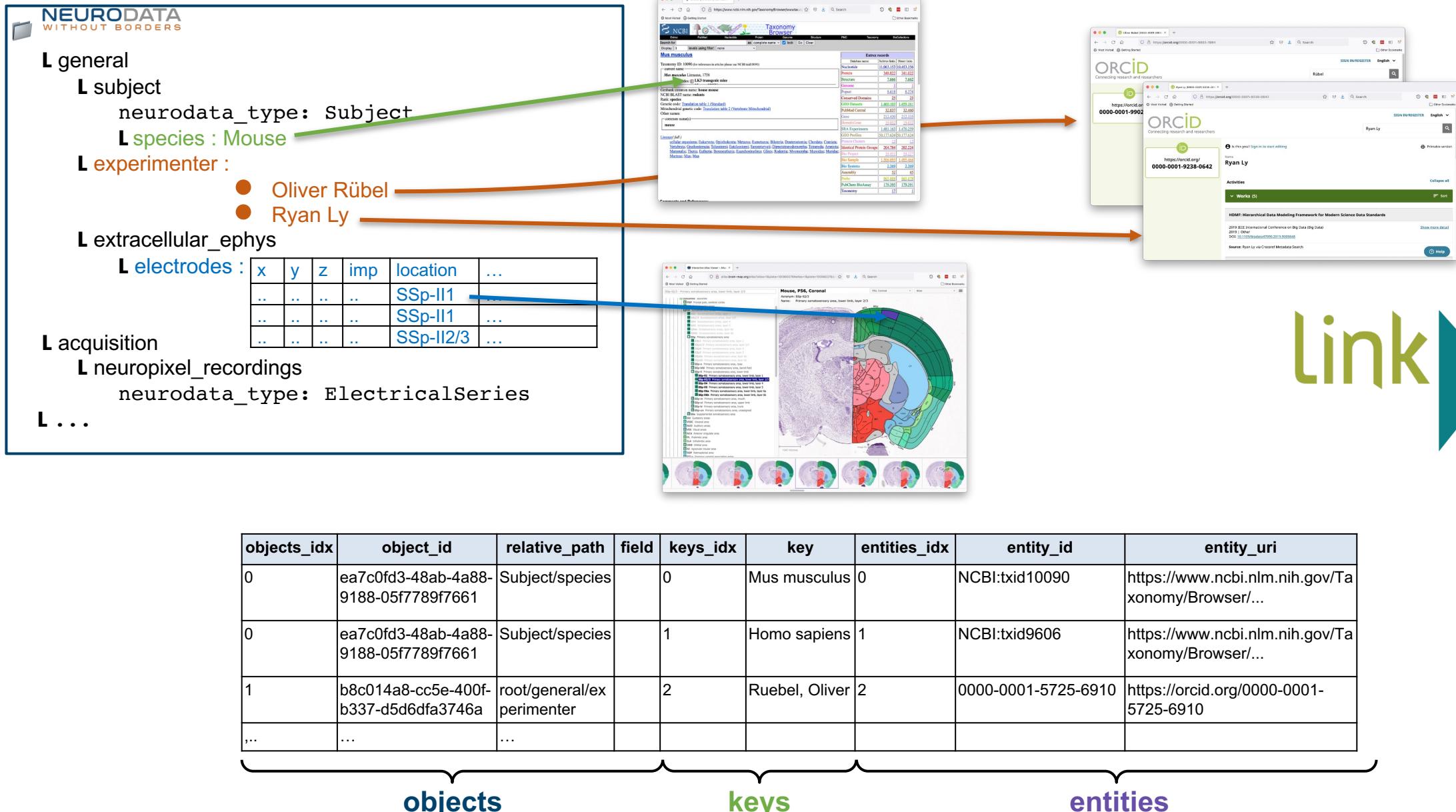
- reductions in cost, time, and effort;
- improved quality, reliability, and accuracy of results
- access to new scientific capabilities

Approach:

- Adapt NWB software to support and integrate with community tools
- Consulting and support for developers
- Dedicated standard and software projects
 - BEADL (behavioral task description language)
 - AqNWB (C++ API for data acquisition)
 - NWB GUIDE user interface for data conversion



NWB + External Resources



42

Develop and maintain an accessible and sustainable open source software ecosystem for NWB

Goal: Ensure readiness of NWB for production use, enhance adoption, and enable the community to contribute to NWB

Software Strategy:

- **Accessibility:** Open source, easy-to-install (pip, conda etc.), detailed documentation, INCF endorsed, R&D100 award
- **Reliability:** Continuous integration, testing, pull requests, and code reviews
- **Stability:** Software management, insulate components, rules and guidelines, regular releases
- **Support:** nwb.org, GitHub, Slack (600+ members), mailing list (>1000 members), Twitter, YouTube, INCF Training Space, ReadTheDocs, newsletter etc.
- **Functionality:** Validator, extensions, support advanced data types, advanced I/O, integration with data libraries (Pandas, numpy etc.)

The collage illustrates the NWB ecosystem through various software interfaces and documentation. It includes:

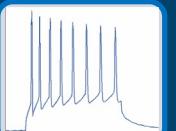
- A 2019 R&D 100 WINNER logo.
- The Neurodata Without Borders website, which is INCF endorsed.
- A GitHub repository page for NeurodataWithoutBorders/pynwb, showing a green 'All checks have passed' status bar with several successful test results listed.
- The NWB website homepage, featuring the Neurodata Without Borders logo and a banner for the 'NWB Virtual User Training Workshop'.

Multidisciplinary team science at work

Applications



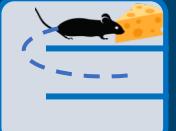
Extracellular
electrophysiology



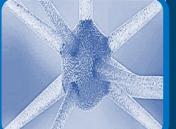
Intracellular
electrophysiology



Optical physiology



Behavior



Simulations

Technology Teams



O. Rübel



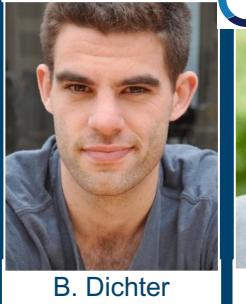
A. J. Tritt



R. Ly



M. Avaylon



B. Dichter



C. Baker



A. Weigl



L. Niu



S. De Vries



A. Buccino



L. Ng



Y.O. Halchenko

NWB Executive Board



K. Bouchard
(LBNL)



B.W. Brunton
(UW)



E. Buffalo
(UW)



A. Churchland
(UCLA)



L. M. Frank
(UCSF)



S. Ghosh
(MIT)



A. Kepcs
(WUSTL)



M. Murthy
(Princeton)



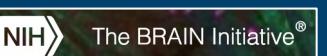
H.D. Mansvelder
(VU Amsterdam)



U. Rutishauser
(Cedars Sinai)

Alumni: L. Ng, C. Koch, F. Sommer, K. Svoboda, M. Meister, K. Amunds

Sponsors



The BRAIN Initiative®



THE KAVLI
FOUNDATION



SIMONS
FOUNDATION

Industry Engagement

- CatalystNeuro
- DataJoint
- Vidrio
- MathWorks
- Kitware

Broader User and Developer Community

Apologies if your name/team is missing!
This slide only shows a very rough cut of some
of the teams and people that work on
developing NWB. Pictures and names of many,
many important members of the NWB
community are missing!



NEURODATA
WITHOUT BORDERS

Visit us at [NWB.org](https://nwb.org) and
nwb-overview.readthedocs.io



BERKELEY LAB
Bringing Science Solutions to the World

Legal



BERKELEY LAB
Bringing Science Solutions to the World

- **Disclaimer:** This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.
- **Copyright notice:** All rights reserved. This manuscript has been authored by an author at Lawrence Berkeley National Laboratory under Contract No. DE-AC02-05CH11231 with the U.S. Department of Energy. The U.S. Government retains, and the publisher, by accepting the article for publication, acknowledges, that the U.S. Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for U.S. Government purposes.