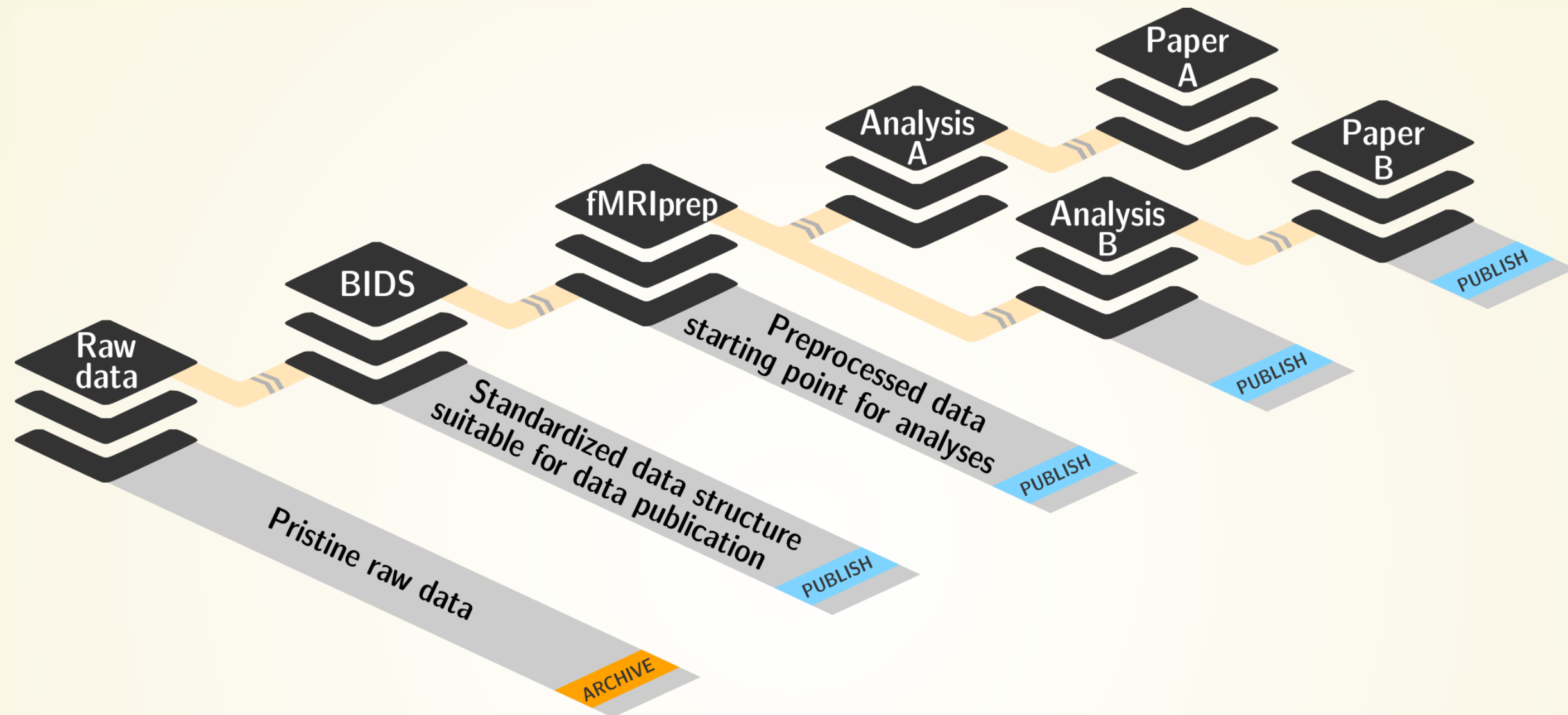


DATASET MANAGEMENT FOR REPRODUCIBILITY & REUSABILITY

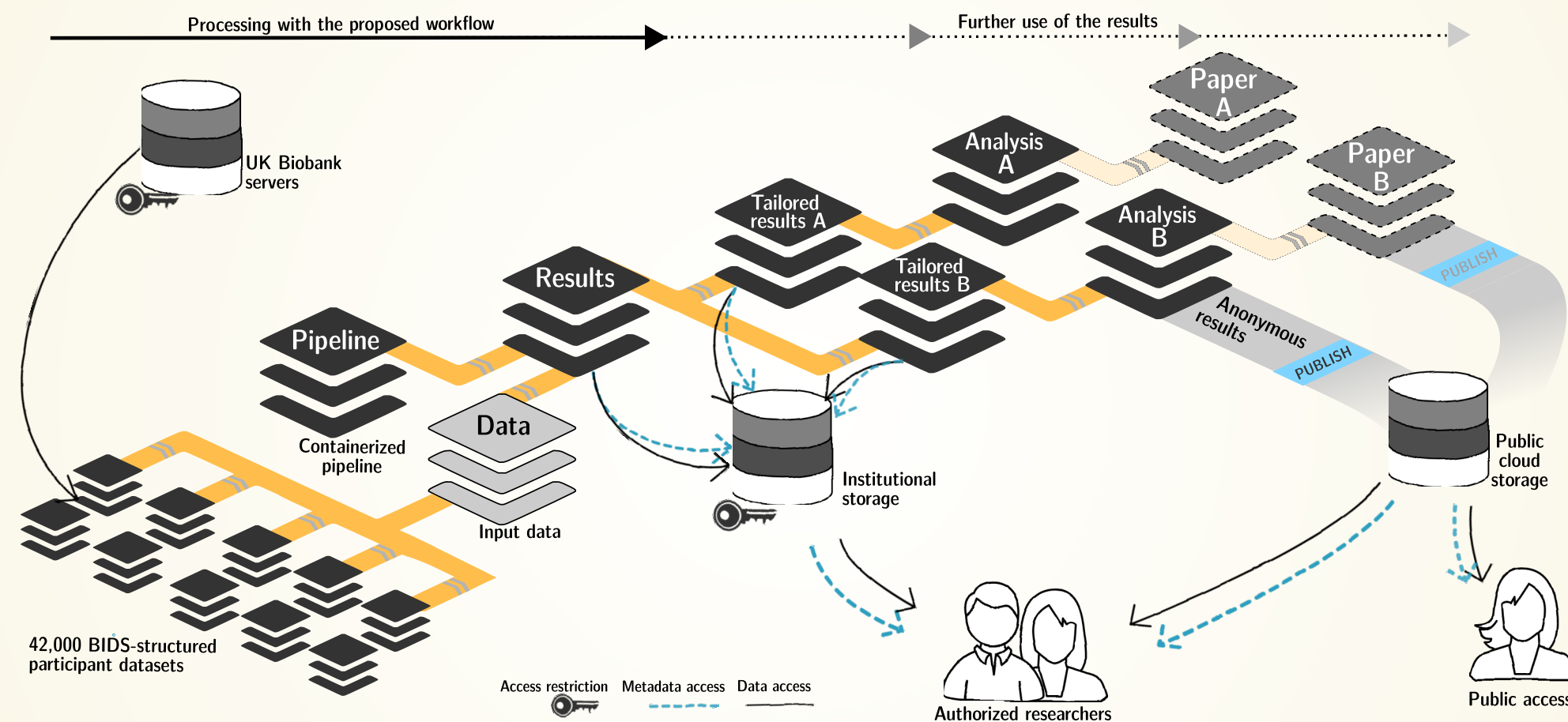
Read more at psychoinformatics-de.github.io/rdm-course/04-dataset-management



When setting up data analyses...

WHY MODULARITY?

- 1. Reuse and access management



- 2. Scalability

```
adina@bulk1 in /ds/hcp/super on git:master> datalad status --annex -r
15530572 annex'd files (77.9 TB recorded total size)
nothing to save, working tree clean
```

(github.com/datalad-datasets/human-connectome-project-openaccess)

WHY MODULARITY?

- 3. Transparency

Original:

```
/dataset
├── sample1
│   └── a001.dat
├── sample2
│   └── a001.dat
└── ...
```

Without modularity, after applied transform (preprocessing, analysis, ...):

```
/dataset
├── sample1
│   ├── ps34t.dat
│   └── a001.dat
├── sample2
│   ├── ps34t.dat
│   └── a001.dat
└── ...
```

Without expert/domain knowledge, no distinction between original and derived data possible.

WHY MODULARITY?

- 3. Transparency

Original:

```
/raw_dataset
├── sample1
│   └── a001.dat
├── sample2
│   └── a001.dat
└── ...
```

With modularity after applied transform (preprocessing, analysis, ...)

```
/derived_dataset
├── sample1
│   └── ps34t.dat
├── sample2
│   └── ps34t.dat
├── ...
└── inputs
    └── raw
        ├── sample1
        │   └── a001.dat
        ├── sample2
        │   └── a001.dat
        └── ...
```

Clearer separation of semantics, through use of pristine version of original dataset within a *new, additional* dataset holding the outputs.

A MACHINE-LEARNING EXAMPLE

Code along or try it later at
handbook.datalad.org/usecases/ml-analysis.html

ANALYSIS LAYOUT

- Prepare an input data set
- Configure and setup an analysis dataset
- Prepare data
- Train models and evaluate them
- Compare different models, repeat with updated data

Imagenette dataset

PREPARE AN INPUT DATASET

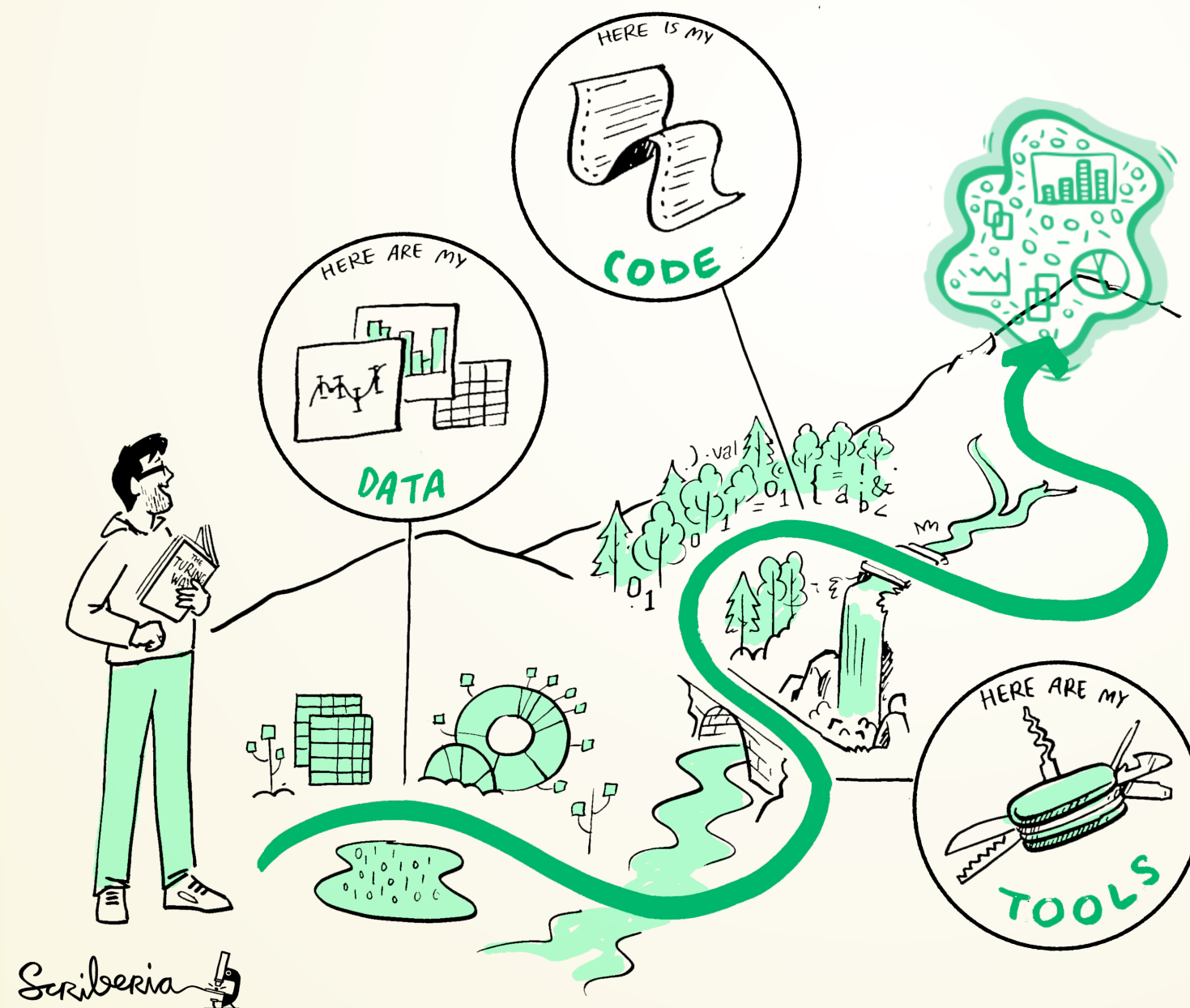
- Create a stand-alone input dataset
- Either add data and `data lad save` it, or use commands such as `data lad download-url` or `data lad add-urls` to retrieve it from web-sources

CONFIGURE AND SETUP AN ANALYSIS DATASET

- Given the purpose of an analysis dataset, configurations can make it easier to use:
 - -c `yoda` prepares a useful structure
 - -c `text2git` keeps text files such as scripts in Git
- The input dataset is installed as a subdataset
- Required software is containerized and added to the dataset

SHARING SOFTWARE ENVIRONMENTS: WHY AND HOW

Science has many different building blocks: Code, software, and data produce research outputs. The more you share, the more likely can others reproduce your results



SHARING SOFTWARE ENVIRONMENTS: WHY AND HOW

- Software can be difficult or impossible to install (e.g. conflicts with existing software, or on HPC) for you or your collaborators
- Different software versions/operating systems can produce different results:

Glatard et al., doi.org/10.3389/fninf.2015.00012

 [About us](#) [Journals](#) [Submit your research](#) [Search](#) [Login](#)

[Frontiers in Neuroinformatics](#) [Articles](#) [Research Topics](#) [Editorial board](#) [About journal](#)

ORIGINAL RESEARCH article

Front. Neuroinform., 24 April 2015
<https://doi.org/10.3389/fninf.2015.00012>

Reproducibility of neuroimaging analyses across operating systems

 [Tristan Glatard \(https://www.frontiersin.org/people/u/156033\)](https://www.frontiersin.org/people/u/156033)^{1,2},
 [Lindsay B. Lewis \(https://www.frontiersin.org/people/u/228462\)](https://www.frontiersin.org/people/u/228462)¹,
 [Rafael Ferreira da Silva \(https://www.frontiersin.org/people/u/156390\)](https://www.frontiersin.org/people/u/156390)³,
 [Reza Adalat \(https://www.frontiersin.org/people/u/158557\)](https://www.frontiersin.org/people/u/158557)¹,
 [Natacha Beck \(https://www.frontiersin.org/people/u/230305\)](https://www.frontiersin.org/people/u/230305)¹,
 [Claude Lepage](#)¹,  [Pierre Rioux](#)¹,

[Download Article](#)

21,348  [73](https://www.altmetric.com/details/103389) (<https://www.altmetric.com/details/103389>)

total views domain=www.frontiersin.org&citati

[View Article Impact \(http://loop-impact.frontiersin.org/10.3389/fninf.2015.00012\)](#)

TABLE OF CONTENTS

Abstract

[Download \(articles/10.3389/fninf.2015.00012/pdf\)](#) [Cookies](#)

1. Introduction
2. Materials and Methods
3. Results
4. Discussion

We use cookies.

Our website uses cookies that are necessary for its operation and to improve your experience. You can review and control your cookies by clicking on "Accept All" or on "Cookies Settings".

SOFTWARE CONTAINERS

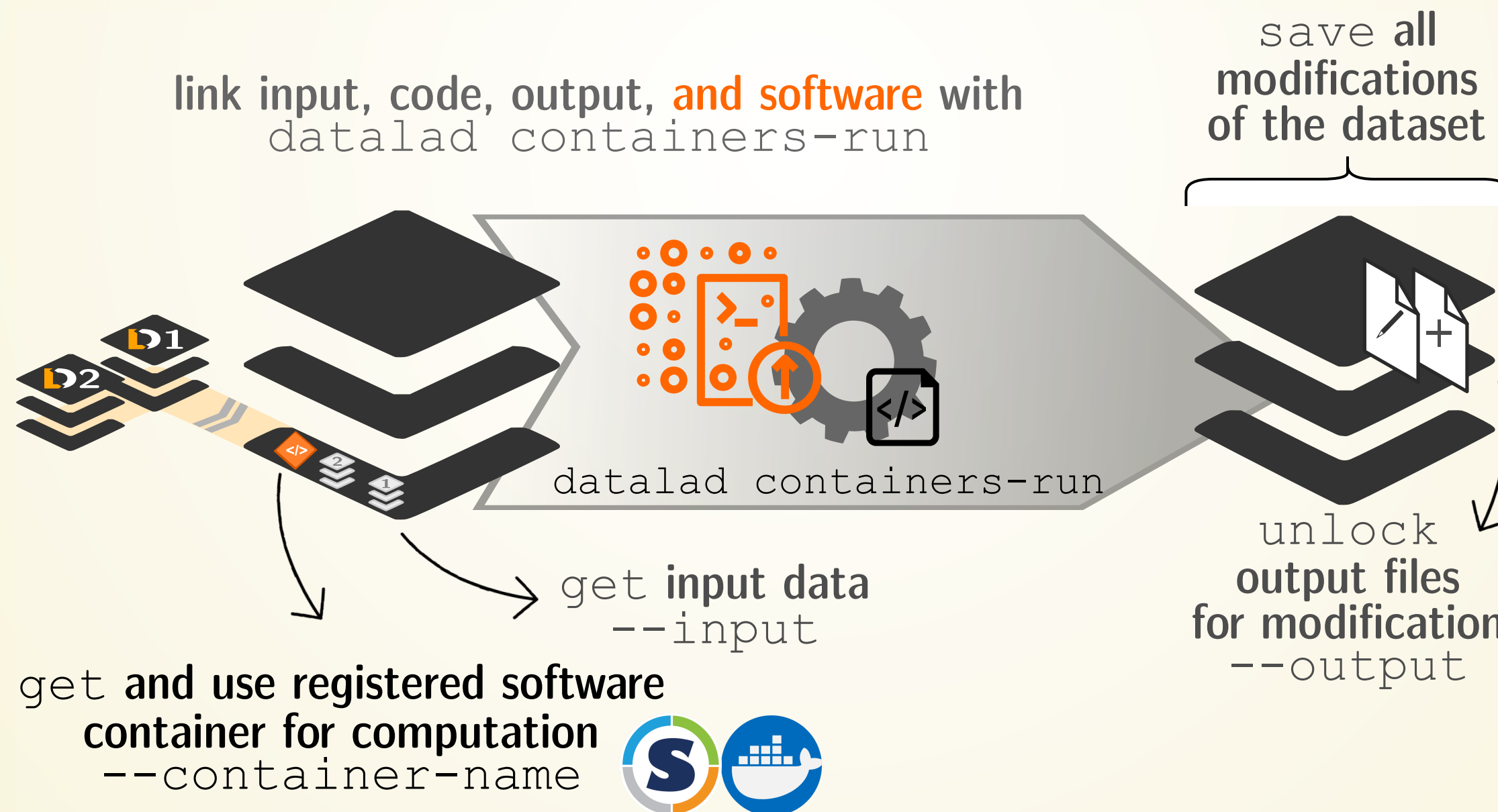
- Put simple, a cut-down virtual machine that is a portable and shareable bundle of software libraries and their dependencies
- **Docker** runs on all operating systems, but requires "sudo" (i.e., admin) privileges
- **Singularity** can run on computational clusters (no "sudo") but is not (well) on non-Linux
- Their containers are different, but interoperable - e.g., Singularity can use and build Docker Images

THE DATALAD-CONTAINER EXTENSION

- The `datalad-container` extension gives DataLad commands to add, track, retrieve, and execute Docker or Singularity containers.

```
pip/conda install datalad-container
```

- If this extension is installed, DataLad can register software containers as "just another file" to your dataset, and **`datalad containers-run`** analysis inside the container, capturing software as additional provenance



DID YOU KNOW...

Helpful resources for working with software containers:

- **repo2docker** can fetch a Git repository/DataLad dataset and builds a container image from configuration files
- **neurodocker** can generate custom Dockerfiles and Singularity recipes for neuroimaging.
- **The ReproNim container collection**, a DataLad dataset that includes common neuroimaging software as configured singularity containers.
- **rocker** - Docker container for R users

PREPARE DATA

- Add a script for data preparation (labels train and validation images)
- Execute it using `data\ad containers - run`

TRAIN MODELS AND EVALUATE THEM

- Add scripts for training and evaluation. This dataset state can be tagged to identify it easily at a later point
- Execute the scripts using `data1ad containers - run`
- By dumping a trained model as a joblib object the trained classifier stays reusable

AND NOW WHAT?

WHEN EVERYTHING IS TRACKED: A REPRODUCIBLE PAPER

Demo: Fully recomputing a real scientific paper (DIY)



- Peer-reviewed paper published in Behavior Research Methods [DOI 10.3758/s13428-020-01428-x]
- Free to reproduce at <https://github.com/psychoinformatics-de/paper-remodnav> more details in the DataLad handbook <http://handbook.datalad.org/r.html?reproducible-paper>.
- Full video: <https://youtube.com/datalad>

ANTICIPATE CHANGE!

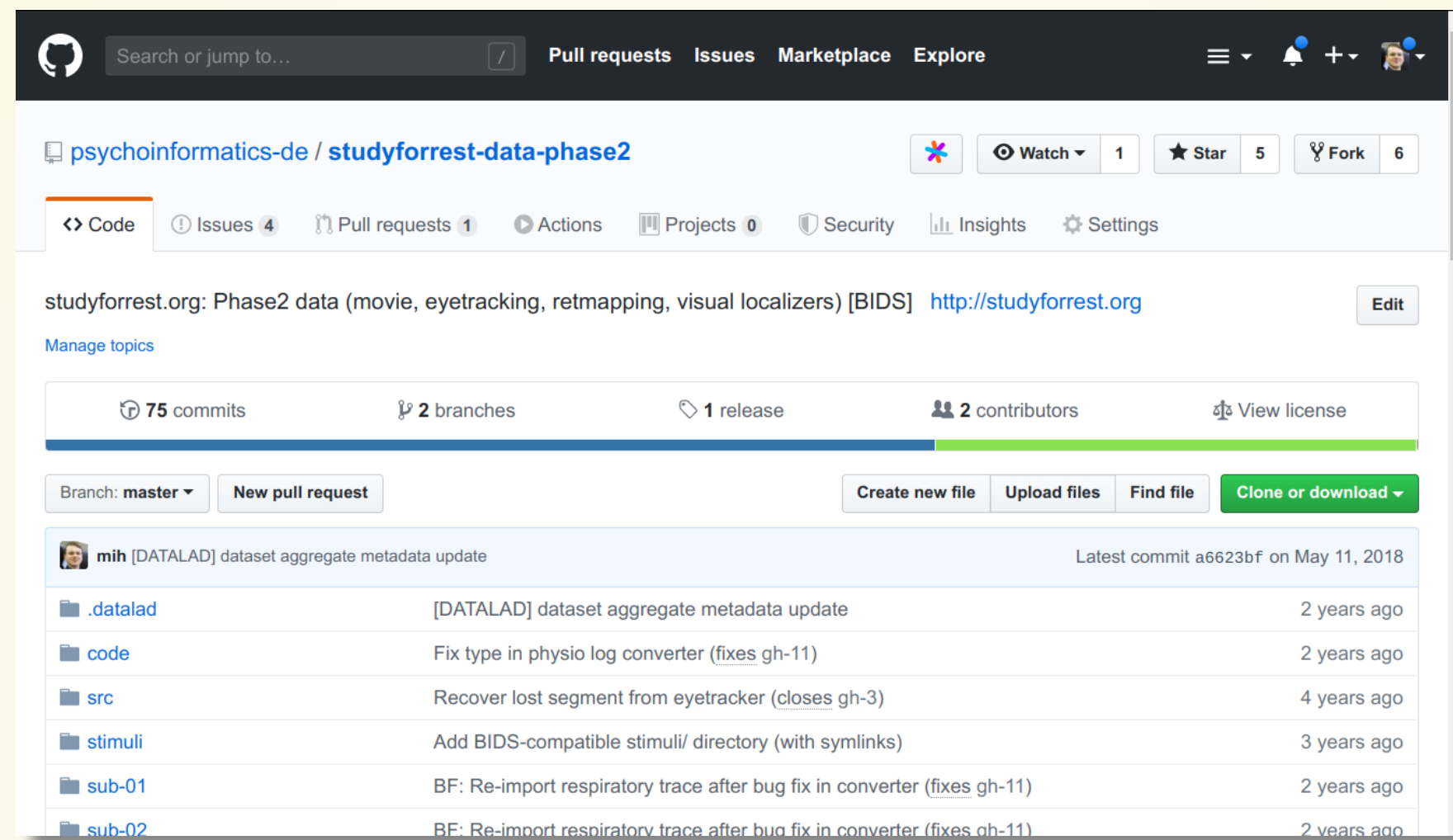
EXHAUSTIVE CAPTURE ENABLES PORTABILITY



Precise identification of data and computational environments, combined for provenance records form a comprehensive and portable data structure, capturing all aspects of an investigation.

Easily take your stuff with you, wherever and whenever you move on!

SERVICES

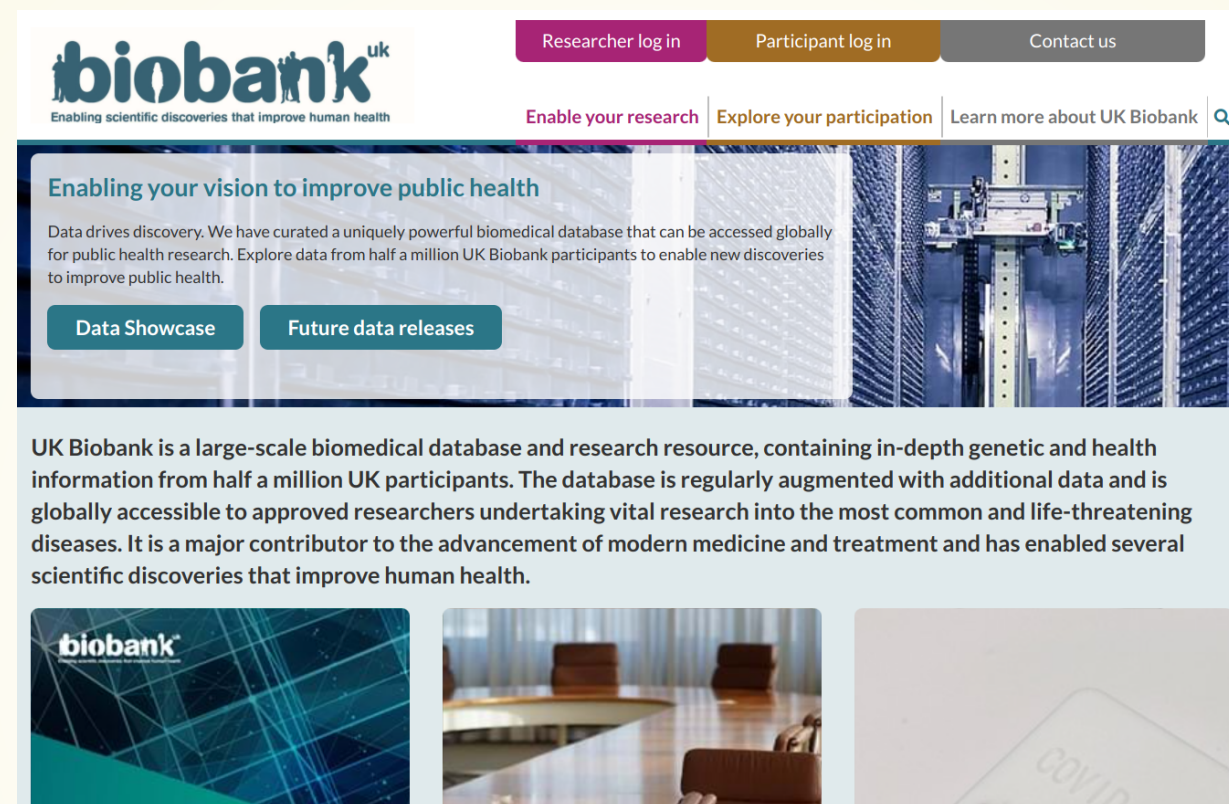


- make *the* difference for advertisement, discovery, convenience
- but imply gigantic dependencies
- often impossible to "take over"

**Make sure data/metadata are self-contained
to facilitate/enable transition to another service**

**IS IT REALLY WORTH THE
INVESTMENT?**

FAIRLY BIG: PROCESS THE UK BIOBANK (IMAGING DATA)

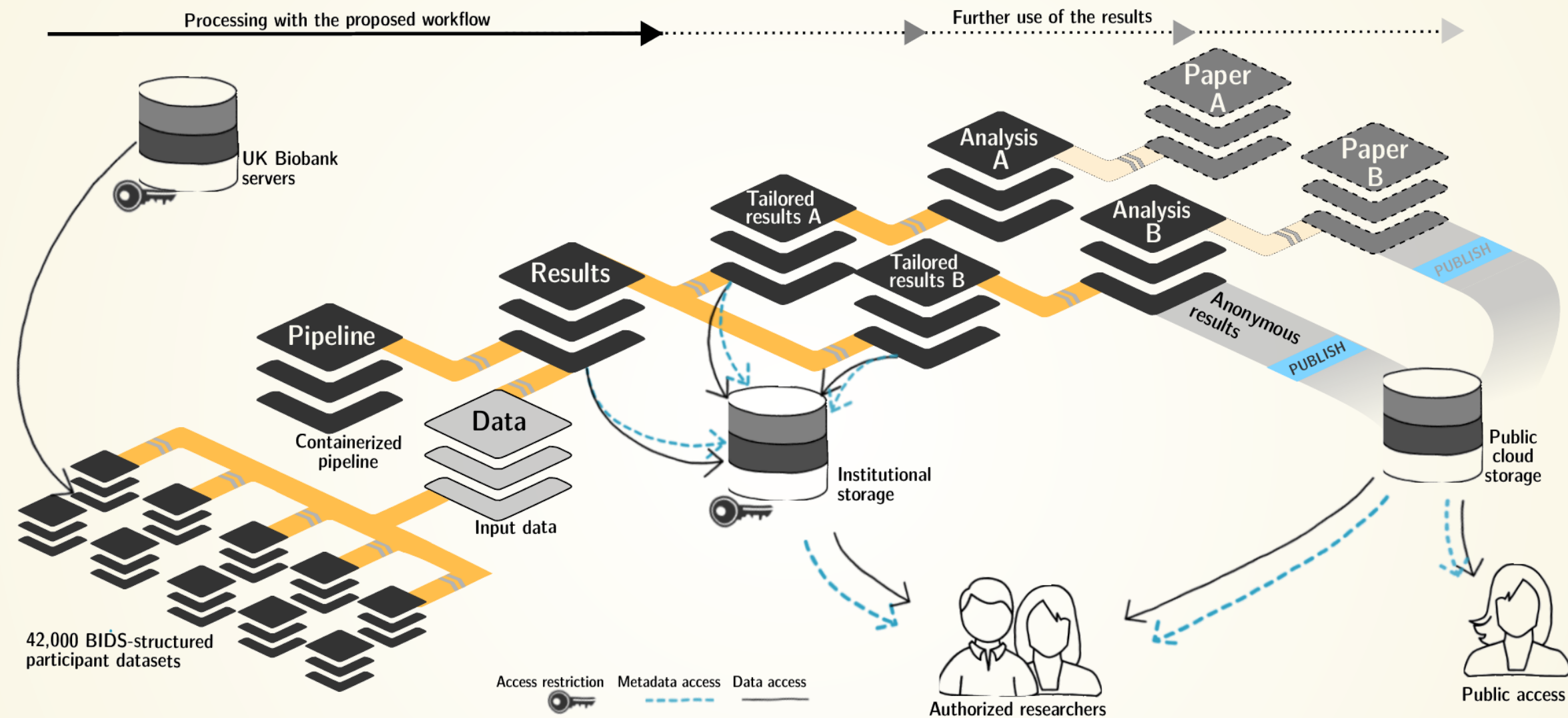


- 76 TB in 43 million files in total
- 42,715 participants contributed personal health data
- Strict DUA
- Custom binary-only downloader
- Most data records offered as (unversioned) ZIP files

CHALLENGES

- Process data such that
 - Results are computationally reproducible (without the original compute infrastructure)
 - There is complete linkage from results to an individual data record download
 - It scales with the amount of available compute resources
- Data processing pipeline
 - Compiled MATLAB blob
 - 1h processing time per image, with 41k images to process
 - 1.2 M output files (30 output files per input file)
 - 1.2 TB total size of outputs

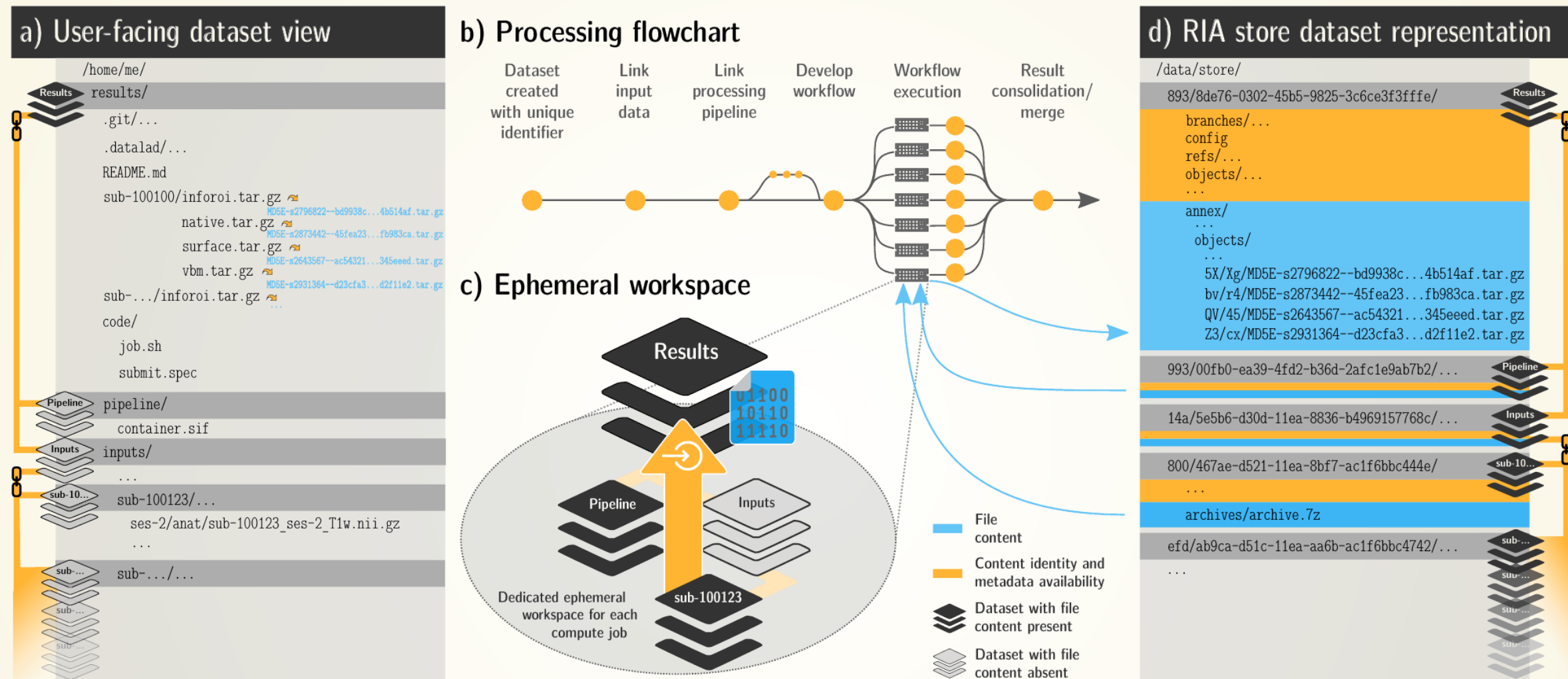
FAIRLY BIG SETUP



- UKB DataLad extension can track the evolution of the complete data release in DataLad datasets
- Full version history
- Native and BIDSified data layout

Wagner, Waite, Wierzba, Hoffstaedter, Waite, Poldrack, Eickhoff, Hanke (2021). FAIRly big: A framework for computationally reproducible processing of large-scale data.

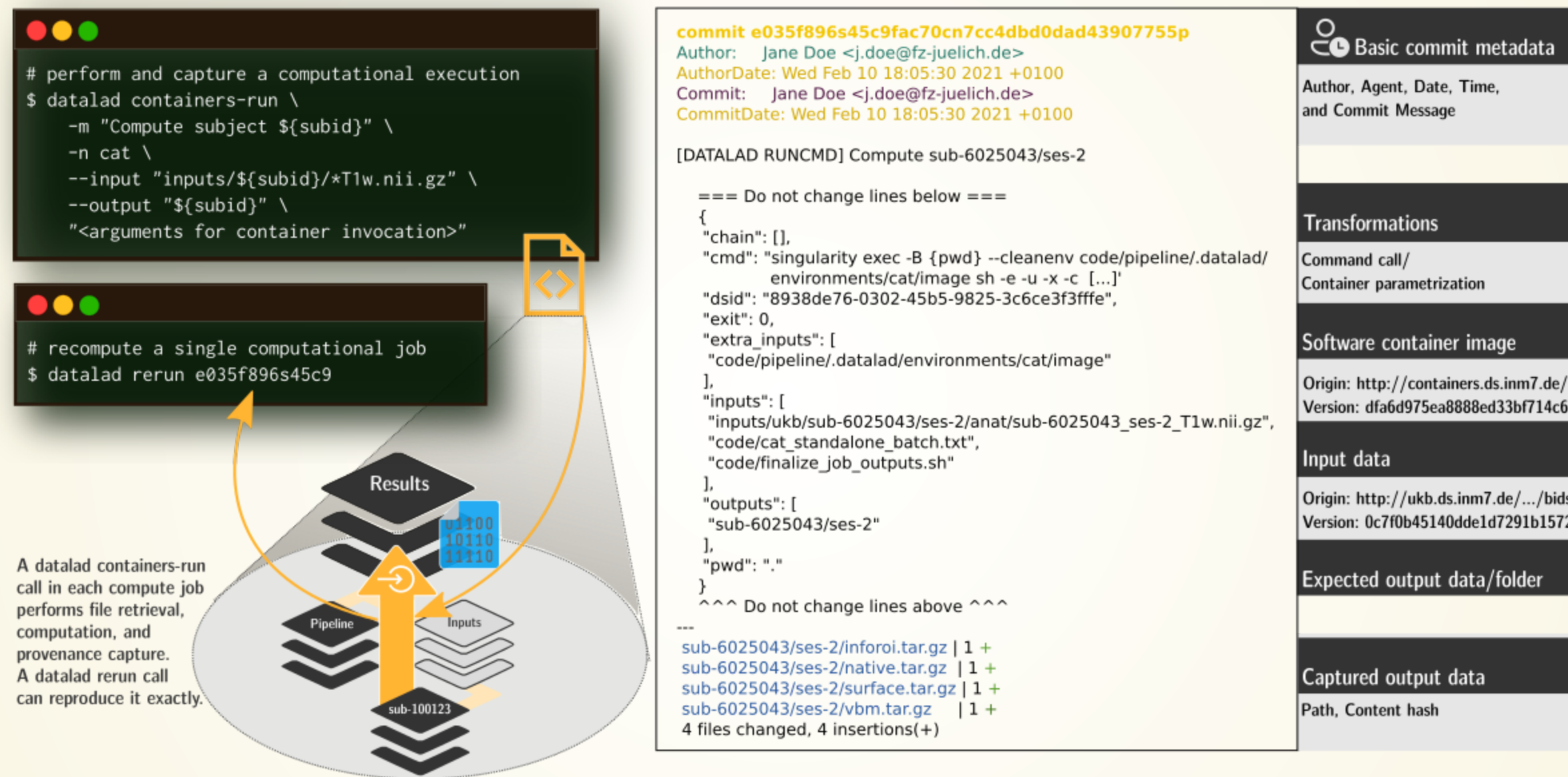
FAIRLY BIG WORKFLOW



- Common data representation in secure environments
 - Content-agnostic persistent (encrypted) storage
 - All computations in freshly bootstrapped ephemeral environments, only using information from a fully self-contained DataLad dataset
- Wagner et al. (2021). FAIRly big: A framework for computationally reproducible processing of large-scale data.

Wagner et al. (2021). FAIRry big: A framework for computationally reproducible processing of large-scale data.

FAIRLY BIG PROVENANCE CAPTURE



- Every single pipeline execution is tracked
- Each execution individually reproducible without HPC access

Wagner et al. (2021). FAIRly big: A framework for computationally reproducible processing of large-scale data.

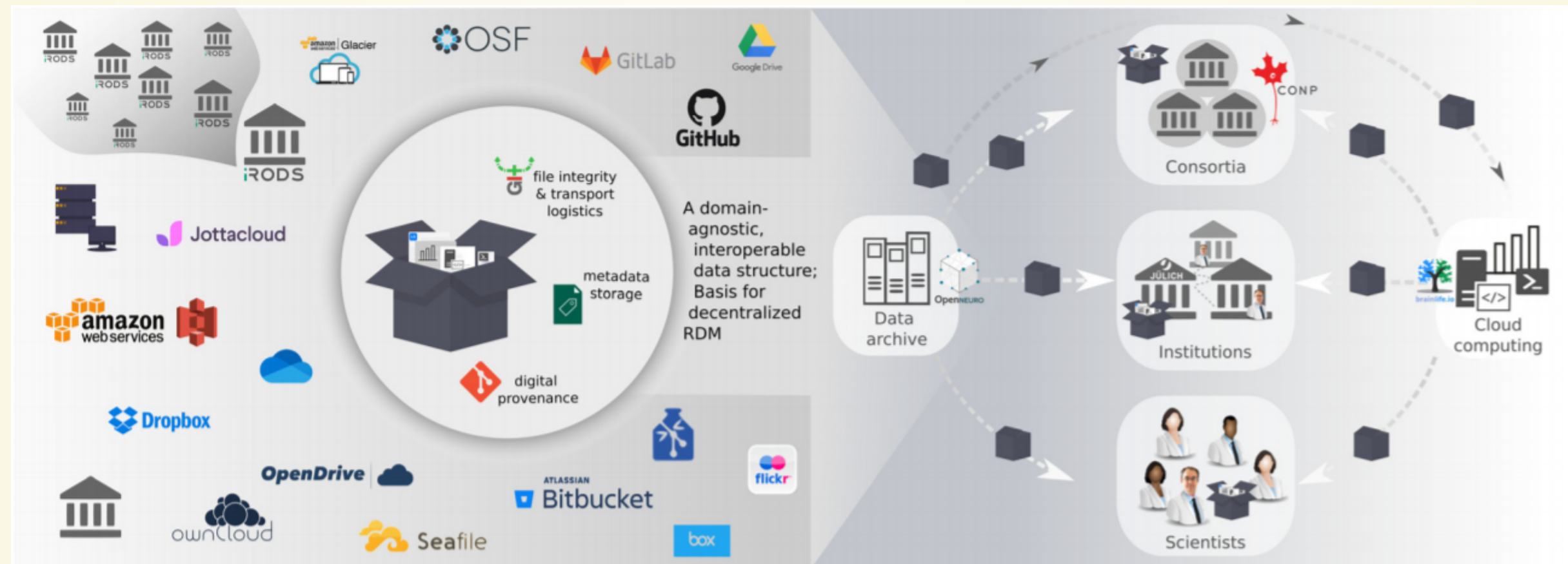
FAIRLY BIG MOVIE

Reproducible processing of 41,180 brain images from the UK Biobank with DataLad



- Rendered exclusively from information captured by DataLad in the output dataset. Full video: <https://youtube.com/datalad>
- Two full (re-)computations, programmatically comparable, verifiable, reproducible -- on any system with data access

INTEROPERABLE DIGITAL RESEARCH ECOSYSTEM



Hanke, Pestilli, Wagner, Markiewicz, Poline & Halchenko (2021). In defense of decentralized research data management. *Neuroforum*, 72, 17-25.

TRAINING AND GUIDELINE DEVELOPMENT

A pragmatic approach to reusable research outputs

Adina S. Wagner¹, J.-B. Poline², Michael Hanke^{1,3}

¹ Institute of Neuroscience and Medicine, Brain & Behaviour INM-7, Research Center Jülich, Germany
² NeuroDataScience - ORIGAMI laboratory, McConnell Brain Imaging Centre, Montreal Neurological Institute-Hospital, Faculty of Medicine and Health Sciences, McGill University, Montreal, Quebec, Canada, and Helen Wills Neuroscience, University of California Berkeley, USA
³ Institute of Systems Neuroscience, Medical Faculty, Heinrich Heine University Düsseldorf, Germany



FAIRly difficult

Science is an incremental process that produces and builds on **more than journal articles**. Code, data, results, or tools of previous finished or unfinished projects (research outputs) fuel new undertakings.

Reusing research objects allows for reproduction, verification, and extending existing work, evidence synthesis, and minimizing duplicate efforts¹. The more reusable outputs are, **at any stage of a project**, the better.



The **FAIR principles**⁴ center around richly curated metadata to reach maximal reusability. In practice, creating **fully FAIR resources is difficult** in systems that yet lack or fail to incentivize the necessary standards and procedures. But **reusability should be improved nevertheless**.

We highlight **four widely accessible strategies** that can elevate reusability as a byproduct of **pragmatic research data management**, even when compliance to FAIR is not yet possible.

REUSABILITY CHECKLIST Can you answer the following questions about your research output?

☐	Are all elements of your project unambiguously identifiable?
☐	Which version of your script was used to generate the output?
☐	What is the precise software environment your analysis was run in?
☐	Exactly what data, at which version, have been used?
☐	Can your project speak for itself?
☐	Is the parametrization of your analysis "codified" and simply structured?
☐	Is relevant processing metadata (e.g. configuration, parametrization) machine-readable, and is used by tools and scripts?
☐	Has relevant process metadata been validated via direct use?
☐	Can you and others infer what has been done to obtain a result?
☐	Are your project's components modular units?
☐	Can someone unfamiliar with your project infer all relevant elements just based on the structure of your project?
☐	Is everything that can be reused set up as a stand-alone entity?
☐	Does your project declare dependencies to all relevant elements of the processing?
☐	Is your project ready to move?
☐	Does your project contain all relevant elements for successful execution (code, data, software environment, ...)?
☐	Do your scripts reference project elements with only relative paths?
☐	Can you move your project to a different computer, or from local to cloud infrastructure, and execute it without adjustments?

Use **version control for code and data**: Git, DataLad, versioning features of cloud services (e.g., AWS). Use them **continuously, not only as a last step prior to publication**.

Ensure that **everything in your project can be versioned**: enshrine software environments in software containers, write your code into **scripts instead of using GUIs**, include all relevant files in your project.

Capture processing parametrization/invoke (in config files/Make-files, bash scripts bundling all execution steps, automatic execution capture) as an entry point into your project!

Ensure that processing relying on this metadata uses it (by reading configurations from files, etc) **in order to test its validity**.

Modularize when files have different ...

- target **audiences** (raw data vs. results),
- **confidentiality** (personal vs. anonymous),
- levels of **accessibility** (proprietary vs. open formats),
- evolutionary **pace** (factual data vs. ongoing study)

Ensure **independence from system idiosyncrasies** (no hard-coded absolute paths, no infrastructure-specific processing components).

Ensure completeness: all required elements should be a part of your project. **Document everything, and, ideally, automate**.

Portability benefits from previous traits!

¹ Mons, B. (2018). *Data Stewardship for Open Science: Implementing FAIR Principles*. CRC Press. ISBN 9780815348184
² Thrun, C. (2017). *Research Data Reusability: Conceptual Foundations, Barriers and Enabling Technologies*. doi.org/10.3390/publications5010002
³ Wilkinson, M. D. et al., (2016). *The FAIR Guiding Principles for scientific data management and stewardship*. doi.org/10.1038/data.2016.18

⁴ Kennedy, D. N., et al. (2019). *Everything Matters: The ReproNim Perspective on Reproducible Neuroimaging*. doi.org/10.3389/fninf.2019.00001
⁵ Hardcastle, T. E. et al. (2018). *Data availability, reusability, and analytic reproducibility*. doi.org/10.1098/rsos.180448
⁶ De Smedt, K. et al. (2020). *FAIR Digital Objects for Science: From Data Pieces to Actionable Knowledge Units*. doi.org/10.3390/publications8020021



Version controlled

All elements involved in the generation of research outputs - from raw or derived data to software - evolve. Changes in these building blocks influence the resulting research output⁴.

Identifying precise inputs of a research output is a prerequisite for retracing, trusting, and reproducing its genesis, and is possible when version control systems unambiguously track digital files.

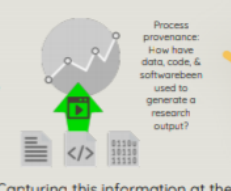


Changed building blocks?

Changed research outputs!

Actionable

The details of how code, software, or data have been used to generate research outputs (process provenance) are volatile, but required to achieve reproducibility⁵.



Capturing this information at the time of its creation, by its expert creators, and basing subsequent processing on it, improves metadata quality and validity even if does not comply to community standards. Ideally, it is simply structured, machine readable, and validated by use. Actionable metadata advances research outputs to FAIR research objects⁶.

Reusable research outputs

FAIR digital objects⁶

Common project files

Modular

Structuring projects into modular units that constitute unambiguous multi-use components (such as raw data, processed data, or software) improves the transparency of a project, and allow individual modules to be recombined and reused easily.

Once you have individual modules, you can declare and link them as dependencies.



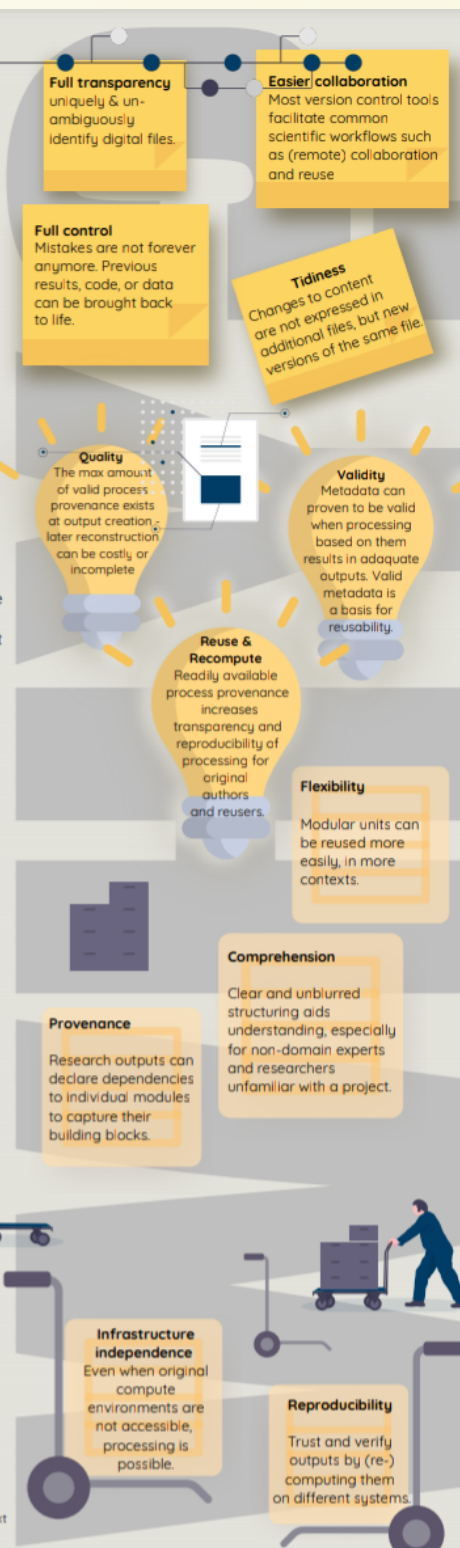
Modules can be reused and their purpose inferred from the structure more easily.

Portable

When your project is portable, it can move between users or infrastructures and run successfully without adjustments. This requires completeness (every required element should be included in the project and identifiable), documentation (how are processing steps executed, what do they do and what do they need), and automation (see Actionable).



Portable projects can run on your laptop, but also your next laptop, your friends workstation, the cluster, or the cloud.



Adina S. Wagner, Jean-Baptiste Poline, Michael Hanke. *A pragmatic approach to reusable research outputs*. 10.7490/f1000research.1118575.1 More hands-on details in the DataLad handbook at <http://handbook.datalad.org>

AFTER THE WORKSHOP

If you have a question after the workshop, you can reach out for help:

Reach out to to the DataLad team via

- [Matrix](#) (free, decentralized communication app, no app needed). We run a weekly Zoom office hour (Thursday, 4pm Berlin time) from this room as well.
- [the development repository on GitHub](#)

Reach out to the user community with

- A question on [neurostars.org](#) with a data lad tag

Find more user tutorials or workshop recordings

- On [DataLad's YouTube channel](#)
- In the [DataLad Handbook](#)
- In the [DataLad RDM course](#)
- In the [Official API documentation](#)