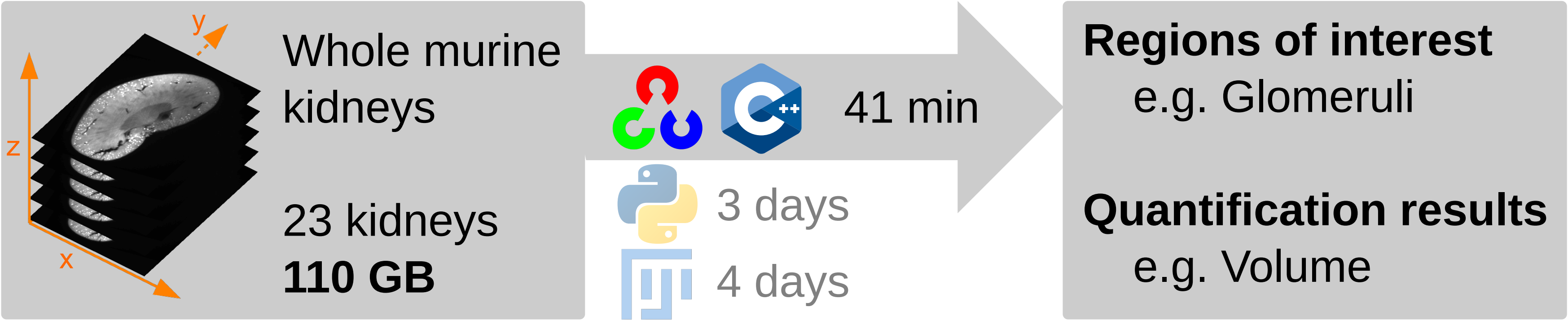
MISA++: a standardized interface for automated high-performance big volume image analysis

Ruman Gerst[1,2], **Anna Medyukhina**[1], **and Marc Thilo Figge**[1,3]

[1] Applied Systems Biology, Leibniz Institute for Natural Product Research and Infection Biology – Hans-Knöll-Institute, Jena, Germany
[2] Faculty of Biological Sciences, Friedrich-Schiller-University Jena, Germany
[3] Institute of Microbiology, Faculty of Biological Sciences, Friedrich-Schiller-University Jena, Germany

1. Implementing a big volume image analysis

Light-sheet fluorescence microscopy (LSFM) allows quantitative three-dimensional analysis of whole organs. This includes the evaluation of structural changes such as a reduced number of glomeruli in kidneys [1].

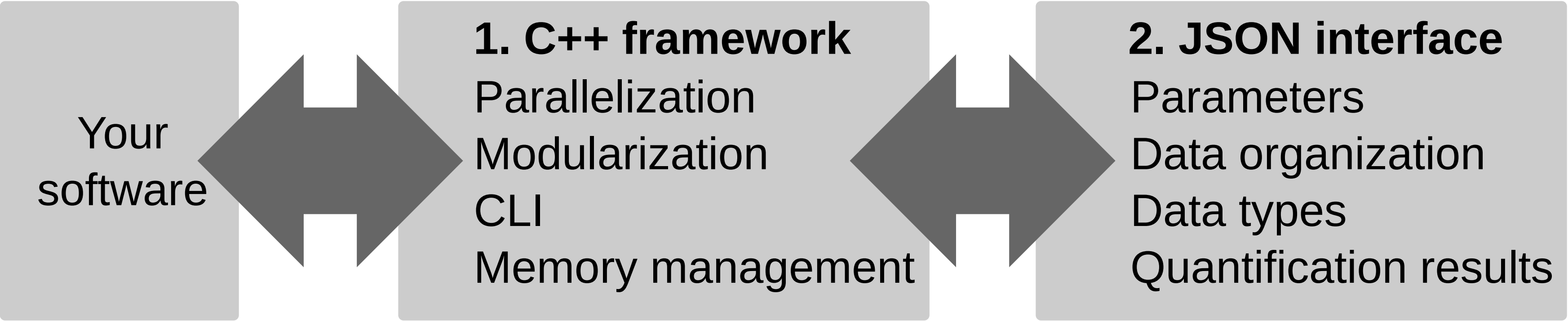


Big volume image data analysis is hardware-intensive and requires high-performance implementations in efficient languages such as C++.

2. MISA++ - A platform for custom C++ analysis tools

MISA++
Modular Image Stack Analysis for C++

Easy development Ready-to-use components for common tasks
Easy integration Standardized JSON interface for data & parameters
Easily extendable Integration of custom data types & more



Comes with: OpenMP OpenCV

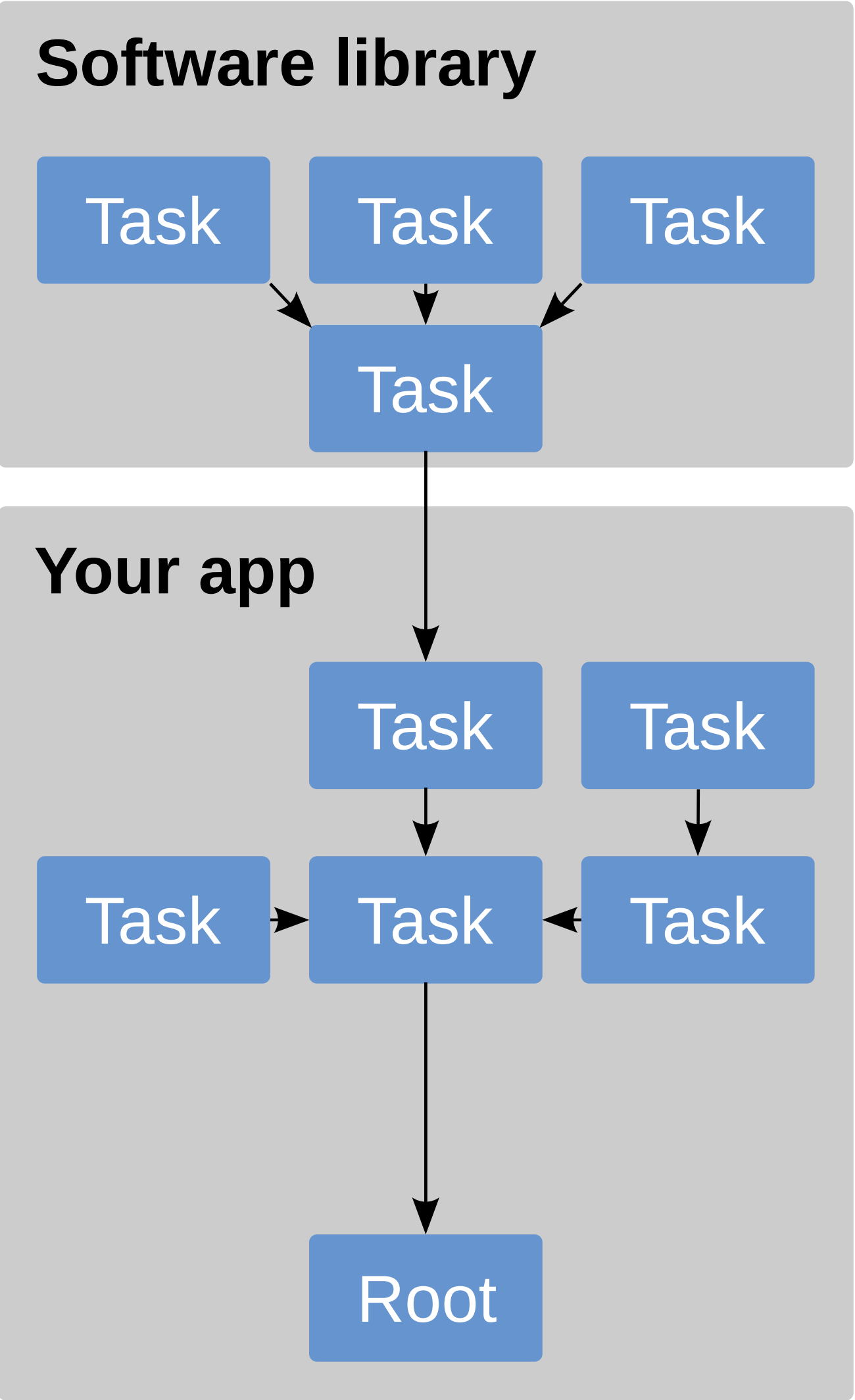
Gerst, R., Medyukhina, A., & Figge, M. T. (2020). MISA++: A standardized interface for automated bioimage analysis. *SoftwareX*, 11, 100405.

3. Parallelization and modularization

Workload is organized in a *directed acyclic graph* (DAG) where nodes represent *tasks* and edges represent dependencies between two tasks.
MISA++ automatically runs tasks with satisfied dependencies in parallel until the work is done.

Easy modularization A set of tasks can be separated into a software library to be developed independently. The DAG of any MISA++ application can be re-used by another application via a standardized modularization interface.

Easy graph construction MISA++ comes with optional functions to simplify DAG creation and enhance code readability.



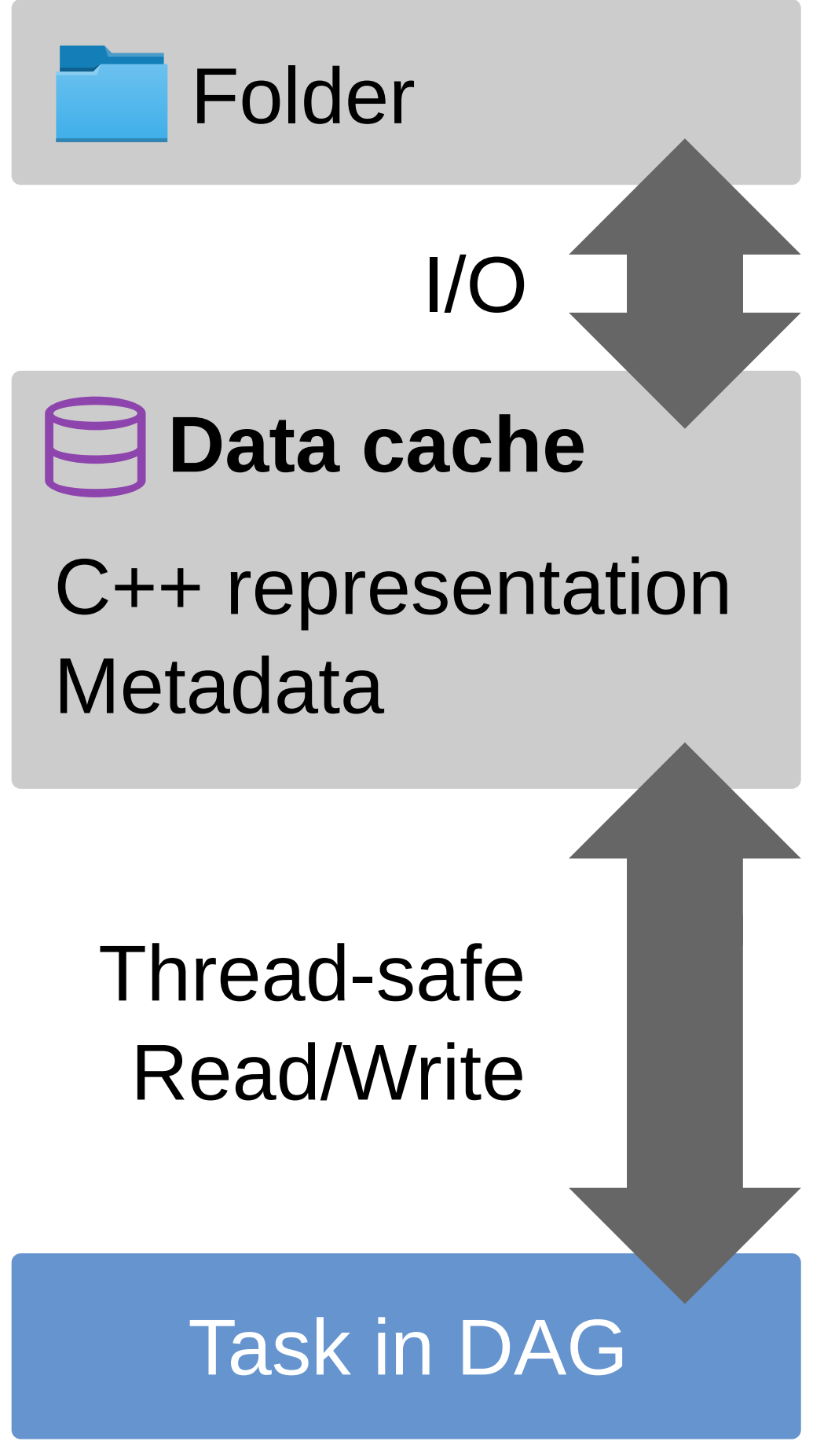
4. Standardized data management

MISA++ wraps custom/third-party data types within structures termed *caches.*

Automated I/O A cache is associated to a location on the hard drive that is used to store currently unused data. On accessing a cache, it automatically loads the data from the hard drive location.

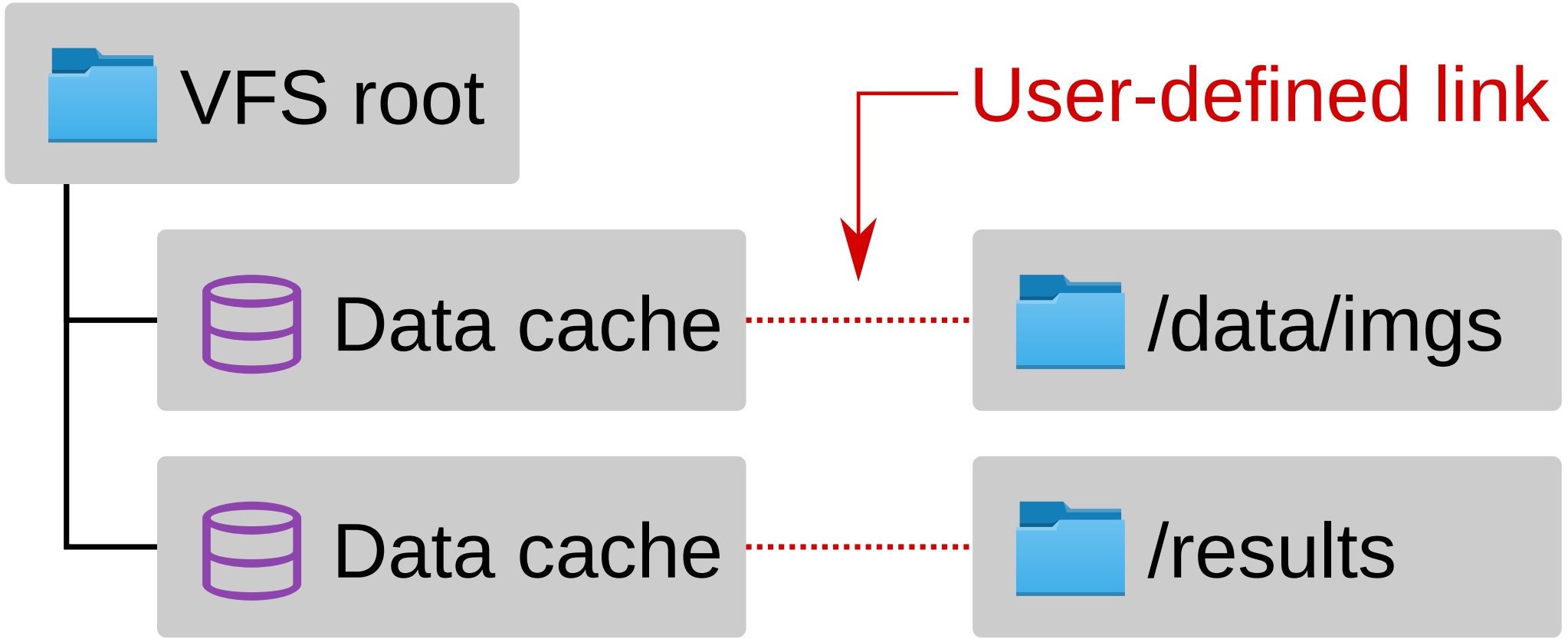
Thread-safe access Tasks can access cached data via thread-safe functions.

Attaching metadata Any cache provides functions to attach user-defined metadata, such as quantification results.



5. Standardized data locations

Caches are associated to a location in a *virtual file system* (VFS) defined in C++ code. VFS locations are linked to hard drive locations by the user.

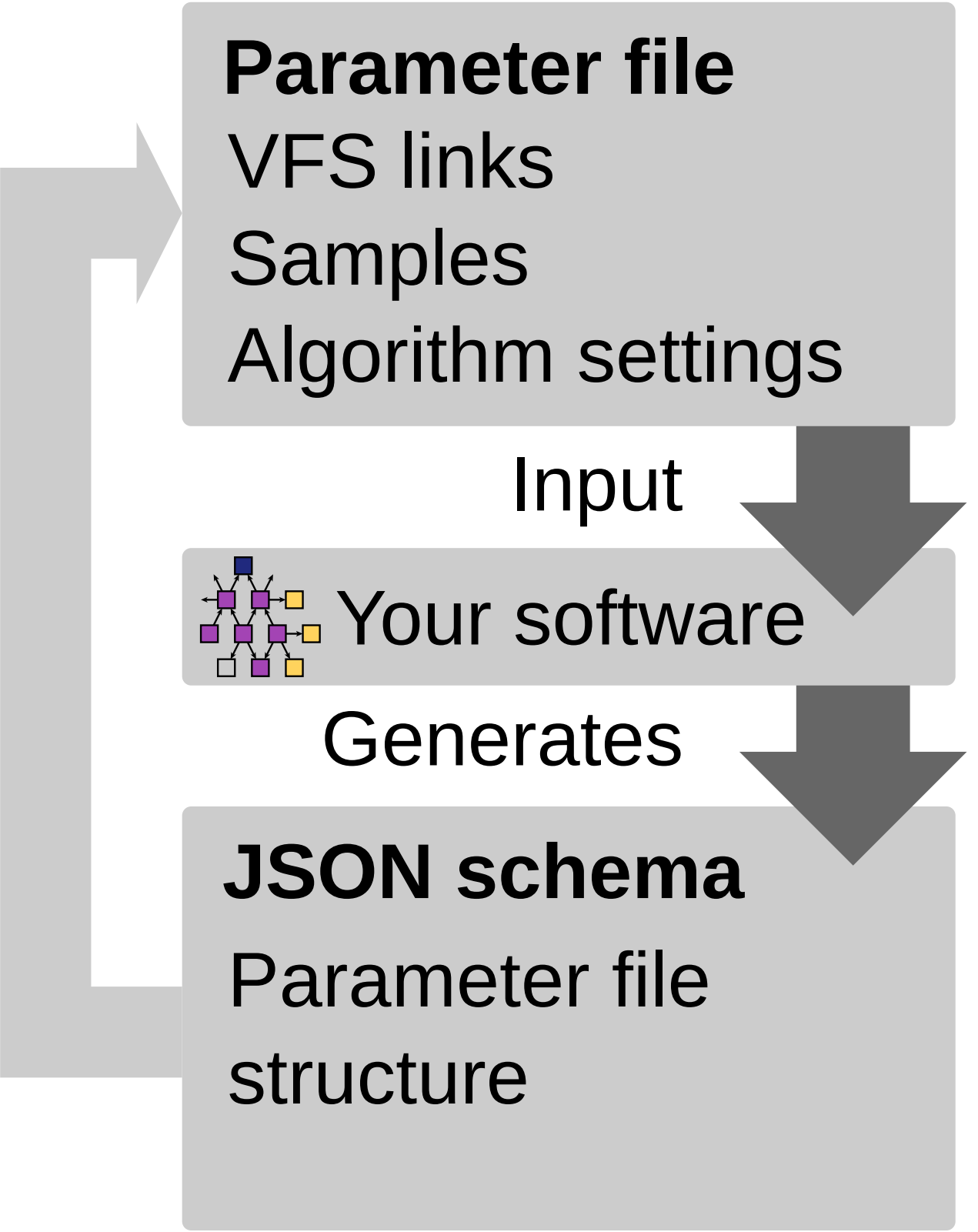


Flexible data locations Users can either follow the VFS structure or redirect cache locations based on the current machine's file system.

6. Standardized parameters & documentation

MISA++ applications read all application settings from a standardized parameter file that links filesystem locations to caches in the VFS, defines samples and their parameters, and allows to change algorithm settings.
Any MISA++ application can automatically generate a file in JSON schema format that describes all available parameter file properties.

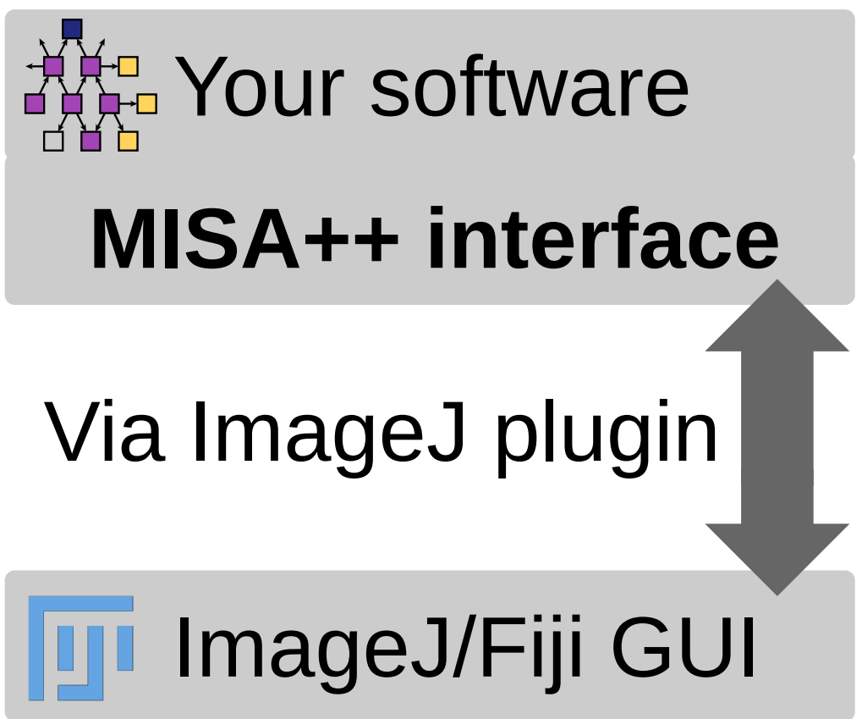
Easy parameter definition Parameters and optional documentation are created in C++ code and automatically assigned to a unique location.



Automated documentation The *JSON schema* describes all parameters, as well as the VFS structure, including input and output data. It can be read by humans and also parsed by other software.

6. Standardized graphical user interface

We developed a graphical user interface for **any** MISA++ application as *ImageJ* plugin.
It extracts all necessary information from the automatically generated *JSON schema* to generate user interfaces for setting up analyses, running them, analyzing results, and creating pipelines.



References:
[1] Anika Klingberg et al., “Fully Automated Evaluation of Total Glomerular Number and Capillary Tuft Size in Nephritic Kidneys Using Lightsheet Microscopy,” Journal of the American Society of Nephrology: JASN 28, no. 2 (February 2017): 452–59, https://doi.org/10.1681/ASN.2016020232.

CONTACT
Ruman.Gerst@leibniz-hki.de
Research Group
Applied Systems Biology