



Research Software Engineering and Software Publication

RWTH Publishing Forum (#10) 25.9.2025

Dr. Anil Yildiz, Chair of Methods for Model-based Development in Computational Engineering, RWTH Aachen University

Dr. Dominik Schmitz, University Library, RWTH Aachen University

Some Housekeeping

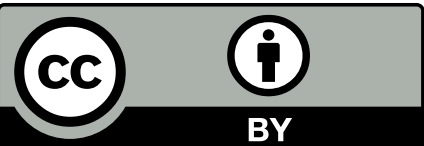
- this session is *not recorded*
- slides will be available on the website of the RWTH University Library (→ Research → Scholarly Publishing → Publishing Forum) and in RWTH Publications
- **Save-the-date: next Publishing Forum**
December 4, 2025, 1:00 p.m. (Zoom):
The insights into the (Open) Leiden Ranking
Mark Neijssel (Centrum voor Wetenschaps- en Technologiestedies (CWTS), Universiteit Leiden)
- Ideas for further topics?
⇒ Let us know! rtp@ub.rwth-aachen.de

Research Software Engineering and Software Publication

Anil Yildiz, Dr. sc.

Methods for Model-based Development in Computational Engineering

RWTH Publishing Forum – 25.09.2025



Models in Science

probing models

phenomenological models

computational models

developmental models

explanatory models

impoverished models

testing models

idealized models

theoretical models

scale models

heuristic models

caricature models

exploratory models

didactic models

fantasy models

minimal models

toy models

imaginary models

mathematical models

mechanistic models

substitute models

iconic models

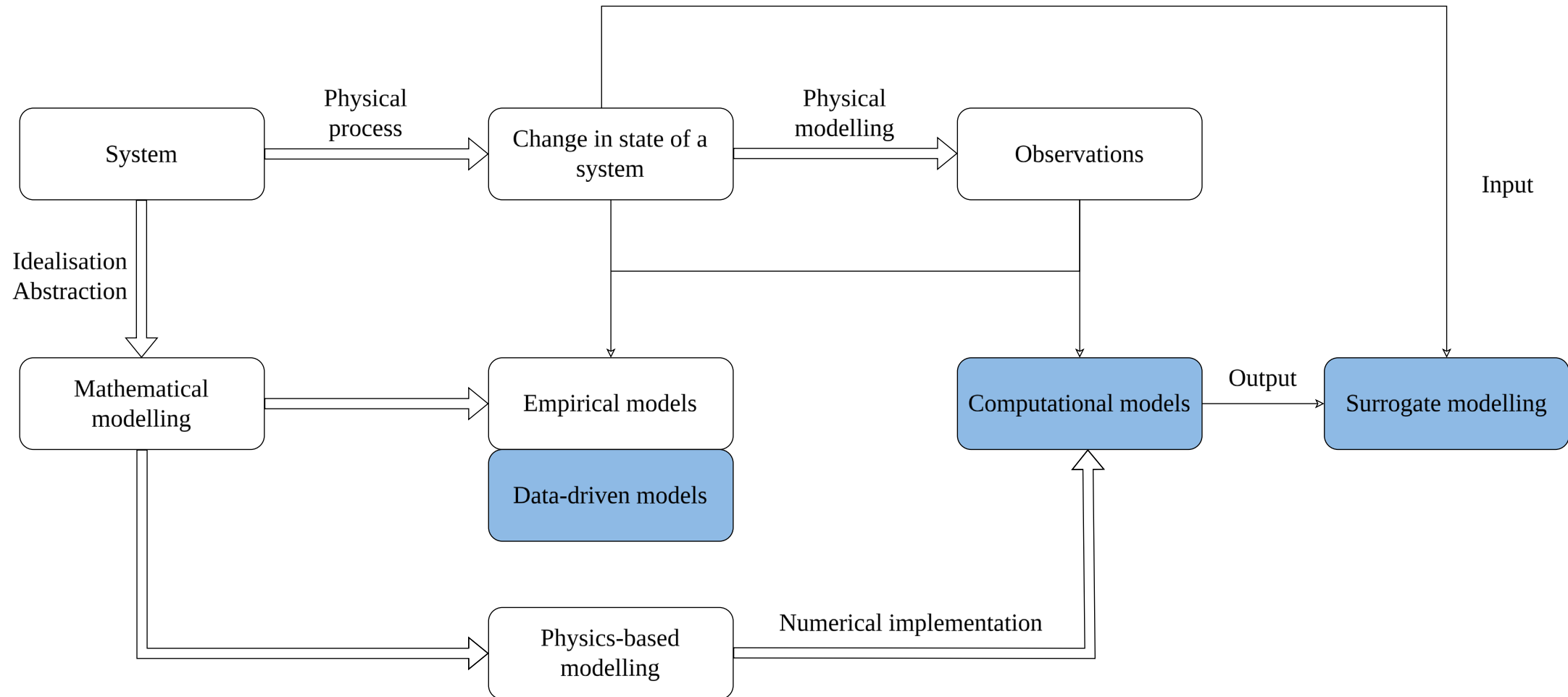
formal models

analogue models

instrumental models

Reference Frigg, Roman, and Stephan Hartmann. 2025. "Models in Science." In *The Stanford Encyclopedia of Philosophy*, edited by Edward N. Zalta and Uri Nodelman

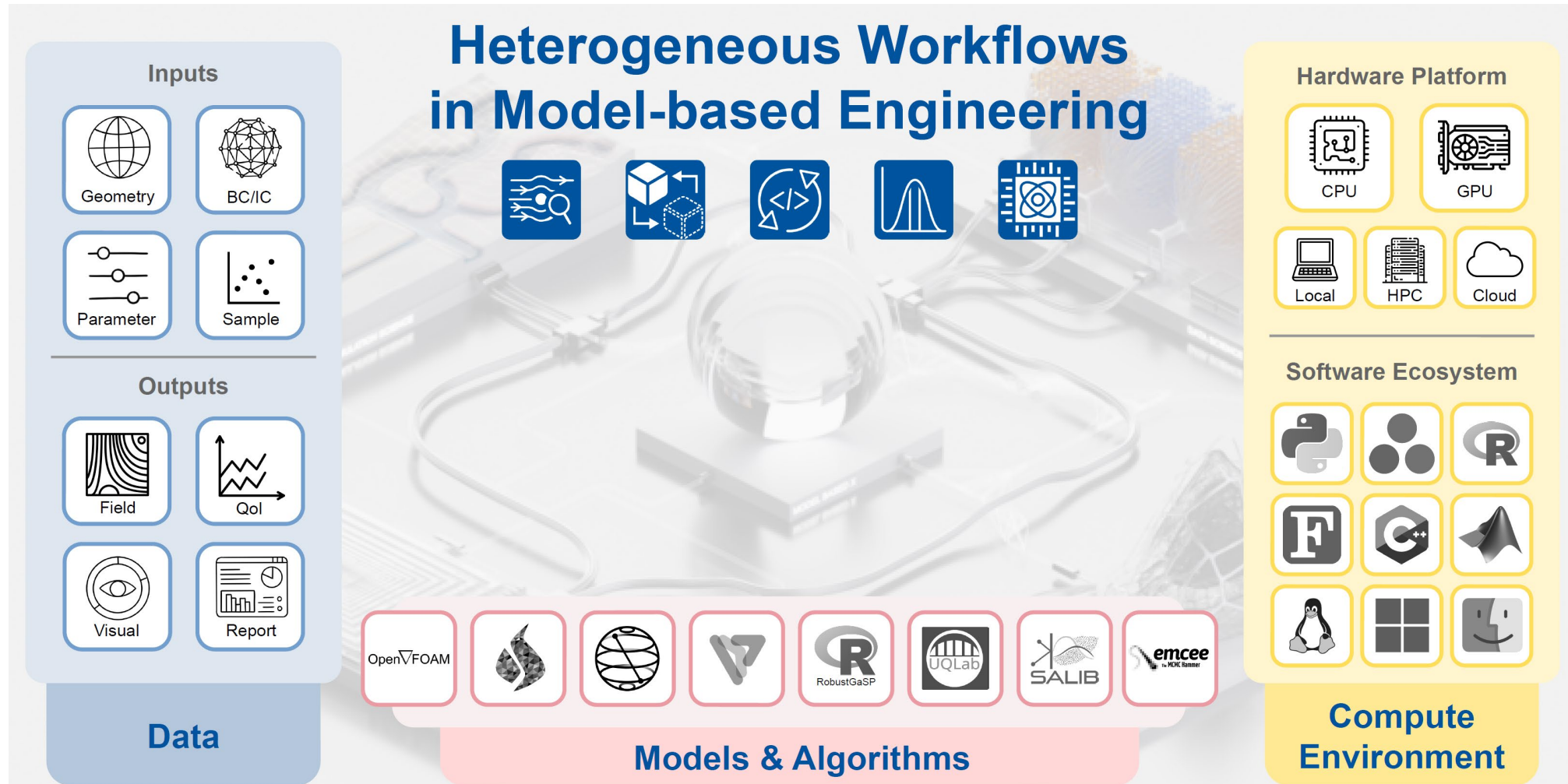
Models in Engineering Applications



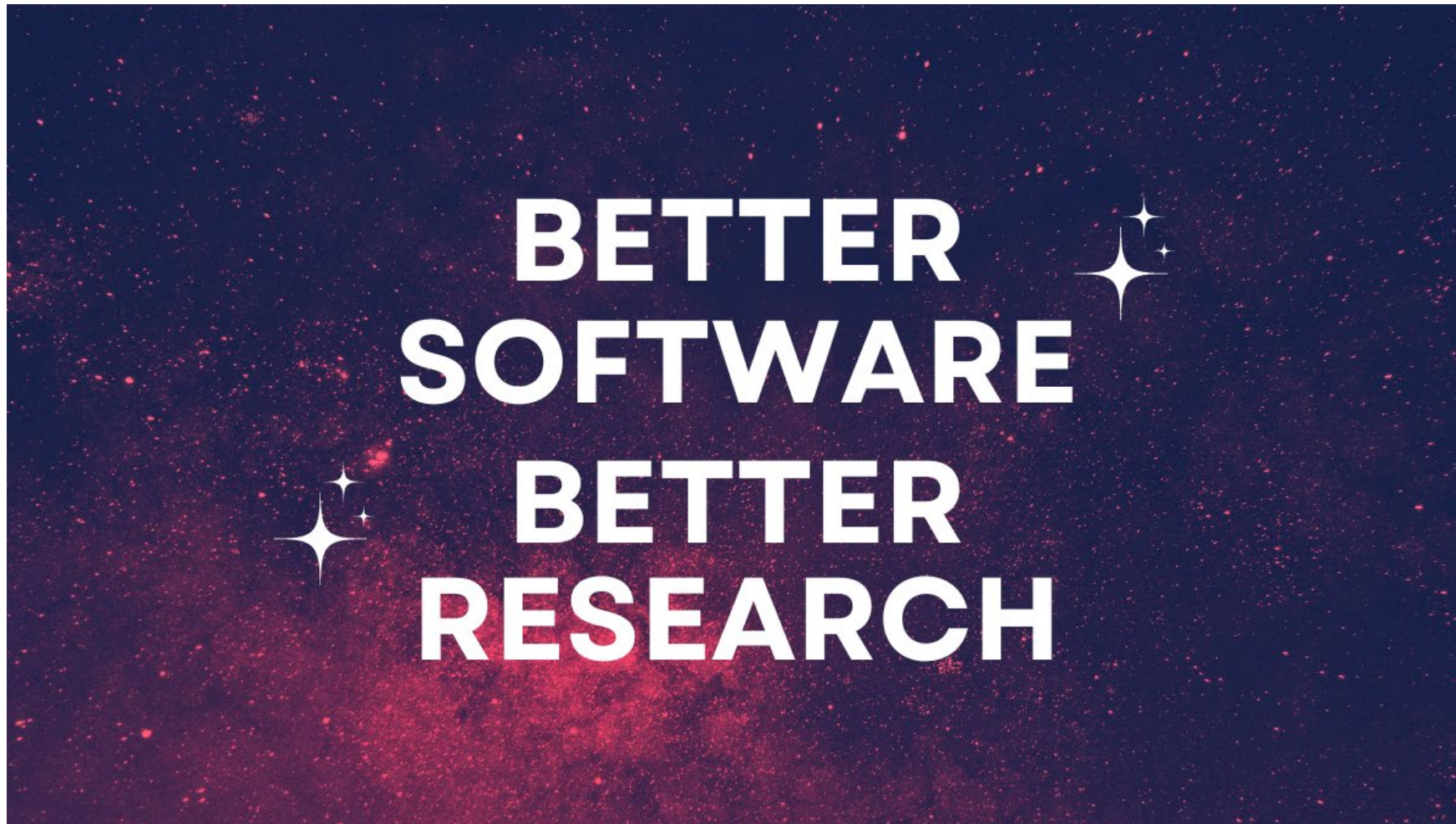
Reference Yildiz, A. 2025. „Model development and classification“. *Sustainable Computational Engineering*. url: mbd.pages.rwth-aachen.de/courses/sce

Software for modelling

- Specialised software (generally commercial) for physics-based simulations, e.g. ABACUS, COMSOL, ANSYS
- Programming languages (Python, MATLAB, C++), often combined with libraries for numerical solvers
- High-throughput tasks, e.g. sensitivity analysis, uncertainty quantification, calibration
- Data management and pre-processing
- (Physics-based/informed) Machine learning algorithms for data-driven modelling
- Optimisation
- Visualisation



Reference Correa et al. 2025. „The I in FAIR: Solutions for Heterogeneous Workflows in Model-based Engineering applications – SHOWME.how “. doi: [10.5281/zenodo.15555565](https://doi.org/10.5281/zenodo.15555565)



Reference Software Sustainability Institute. 2025. url: software.ac.uk. License: CC BY-NC 2.5

Research software engineering

Definitions and parts

the use of software engineering practices, methods and techniques for research software, i.e. software that was made for and is mainly used within research projects

Wikipedia

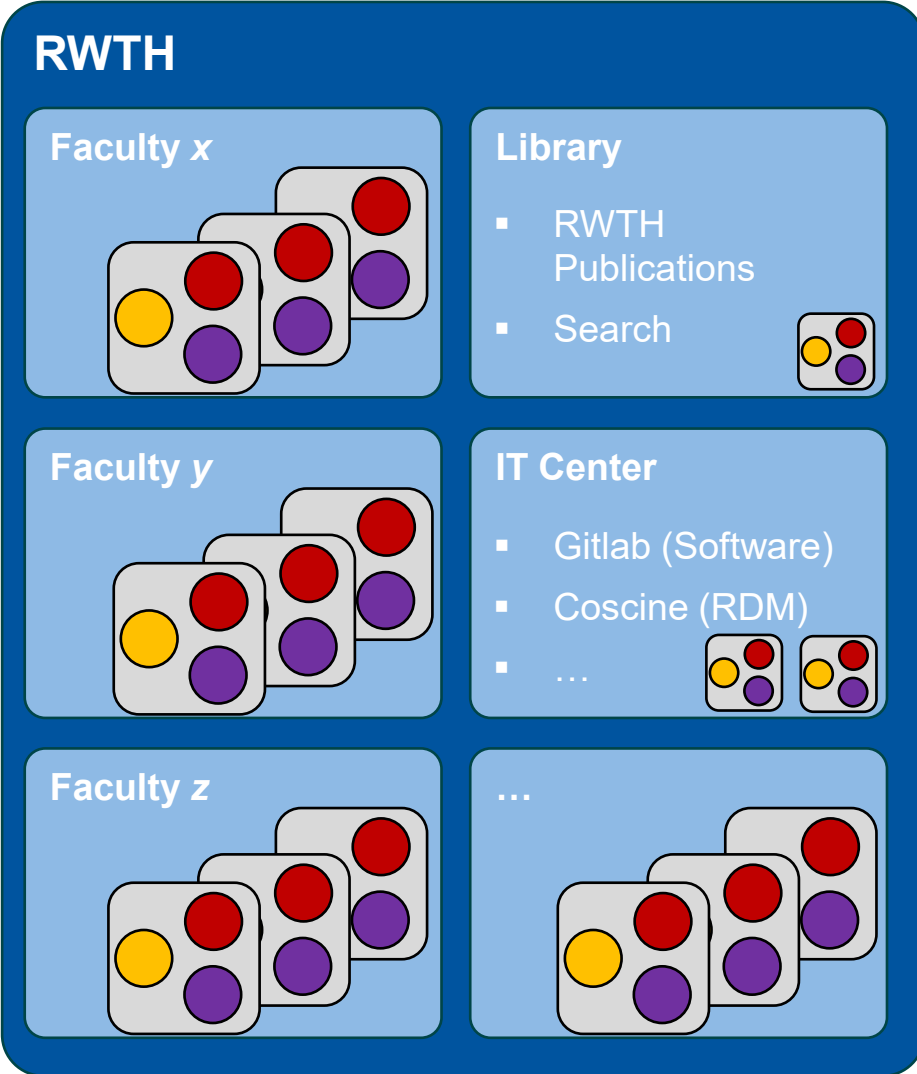
- App infrastructure
- User interactions
- Visualization
- Communication between computing and storage nodes
- Providing web services
- Rights and roles management for access and prevention of undetected changes
- Technical monitoring
- Testing
- Installation, deployment and orchestration

Reference Rumpe et al. 2025. „Research Software Engineering“. url: se-rwth.de/essay/Research-Software-Engineering/

What do we want?

- **Accessibility:** Provide access to source code perpetually
- **Traceability:** Trace and recall any step in development history
- **Ease-of-use:** Take up in new projects and ramp up new employees
- **Maintainability:** Clear structure and modular components promote better testing and debugging, and, in turn, longevity
- **Reproducibility:** Consistently reuse code for simulations and analyses across projects or institutions
- **Reusability:** Extend existing engineering or simulation classes for new applications
- **Citability:** Our intellectual and scholar product can be cited by further users

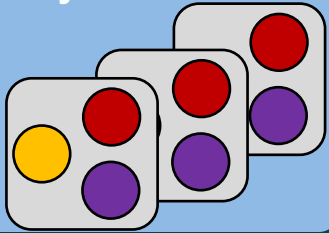
Stakeholders of Software Development at RWTH



Motivation for publishing software

RWTH

Faculty x

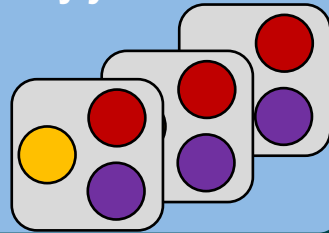


Library

- RWTH Publications
- Search

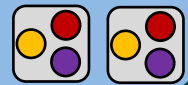


Faculty y

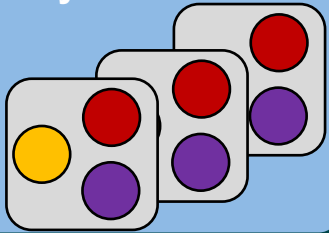


IT Center

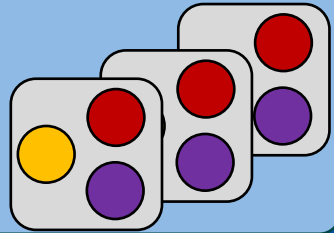
- Gitlab (Software)
- Coscine (RDM)
- ...



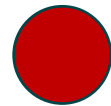
Faculty z



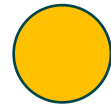
...



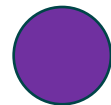
Demonstration of research activity and impact, sustainability of funding, synergies, attractor to the public



Visibility, recognition, reuse of group results, track record for funding



Reproducibility of the singular project results, requirements of funding agencies, expendability into new projects



Visibility of individuals, preparation for a career

Research software engineering

Issues

- Limited availability and usability of research software;
- Redundant and parallel software developments;
- Insufficient skills in software engineering;
- Lack of quality standards for the development and review of research software;
- Lack of reproducibility;
- Lack of recognition of research software development as scientific contribution;
- Missing link between research software development and the scientific reputation system;
- Unclear rules for publication with regard to licenses and intellectual property (IP).

Reference de-RSE e.V. - Society for Research Software. 2025. Research Software Engineers (RSEs) - The people behind research software. url: de-rse.org/en/

Research software engineering

Issues

- Limited availability and usability of research software;
- Redundant and parallel software developments;
- Insufficient skills in software engineering;
- Lack of quality standards for the development and review of research software;
- Lack of reproducibility;
- Lack of recognition of research software development as *scientific* contribution;
- Missing link between research software development and the scientific reputation system;
- Unclear rules for publication with regard to licenses and intellectual property (IP).

Findability

Reference de-RSE e.V. - Society for Research Software. 2025. Research Software Engineers (RSEs) - The people behind research software. url: de-rse.org/en/

Research software engineering

Issues

- Limited availability and usability of research software;
- Redundant and parallel software developments;
- Insufficient skills in software engineering;
- Lack of quality standards for the development and review of research software;
- Lack of reproducibility;
- Lack of recognition of research software development as *scientific* contribution;
- Missing link between research software development and the scientific reputation system;
- Unclear rules for publication with regard to licenses and intellectual property (IP).

RSE
Best Practices

Reference de-RSE e.V. - Society for Research Software. 2025. Research Software Engineers (RSEs) - The people behind research software. url: de-rse.org/en/

Research software engineering

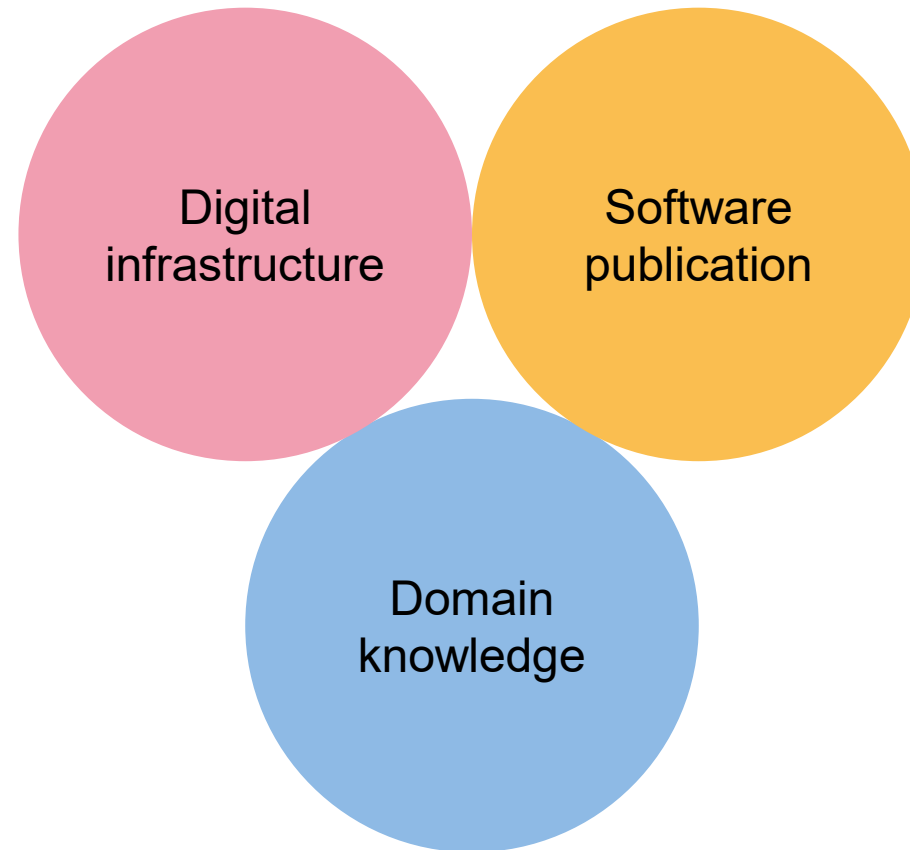
Issues

- Limited availability and usability of research software;
- Redundant and parallel software developments;
- Insufficient skills in software engineering;
- Lack of quality standards for the development and review of research software;
- Lack of reproducibility;
- Lack of recognition of research software development as *scientific* contribution;
- Missing link between research software development and the scientific reputation system;
- Unclear rules for publication with regard to licenses and intellectual property (IP).

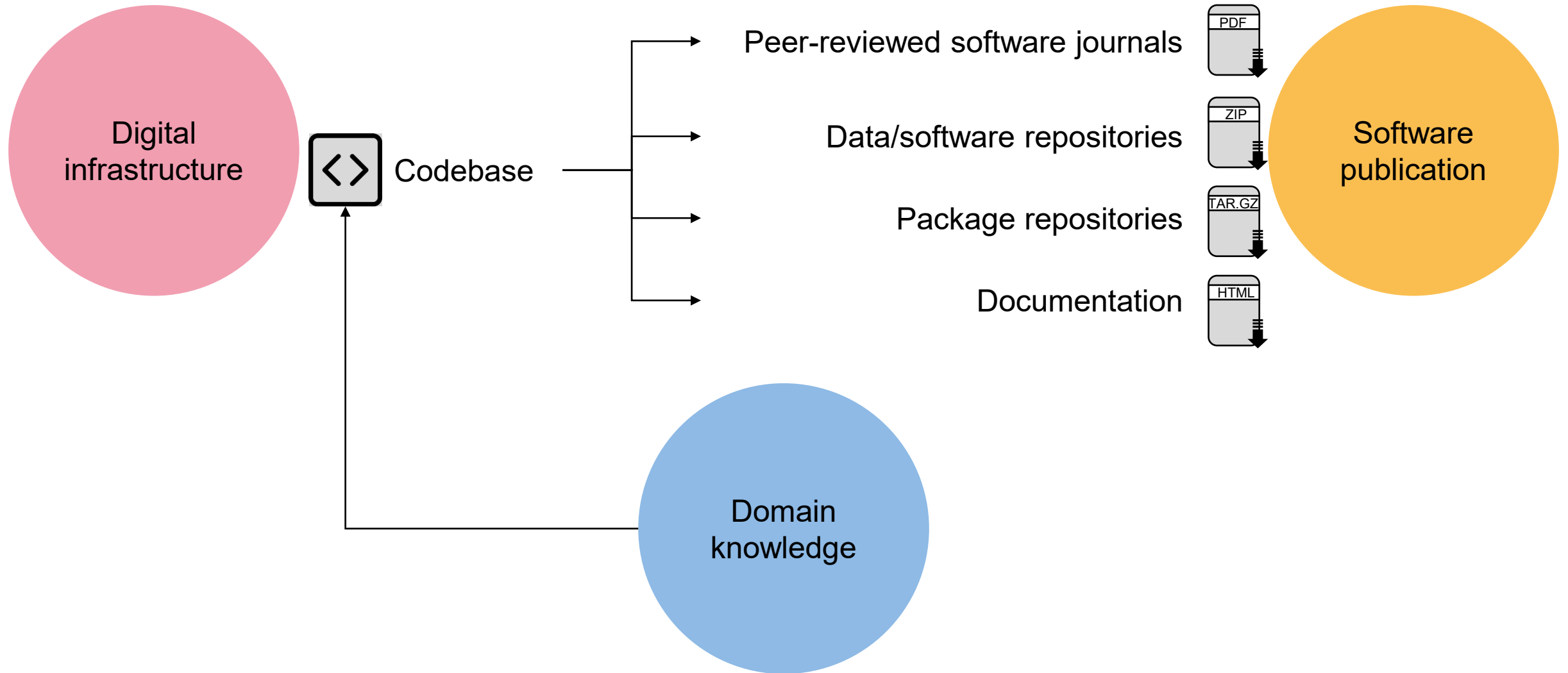
Software
publications

Reference de-RSE e.V. - Society for Research Software. 2025. Research Software Engineers (RSEs) - The people behind research software. url: de-rse.org/en/

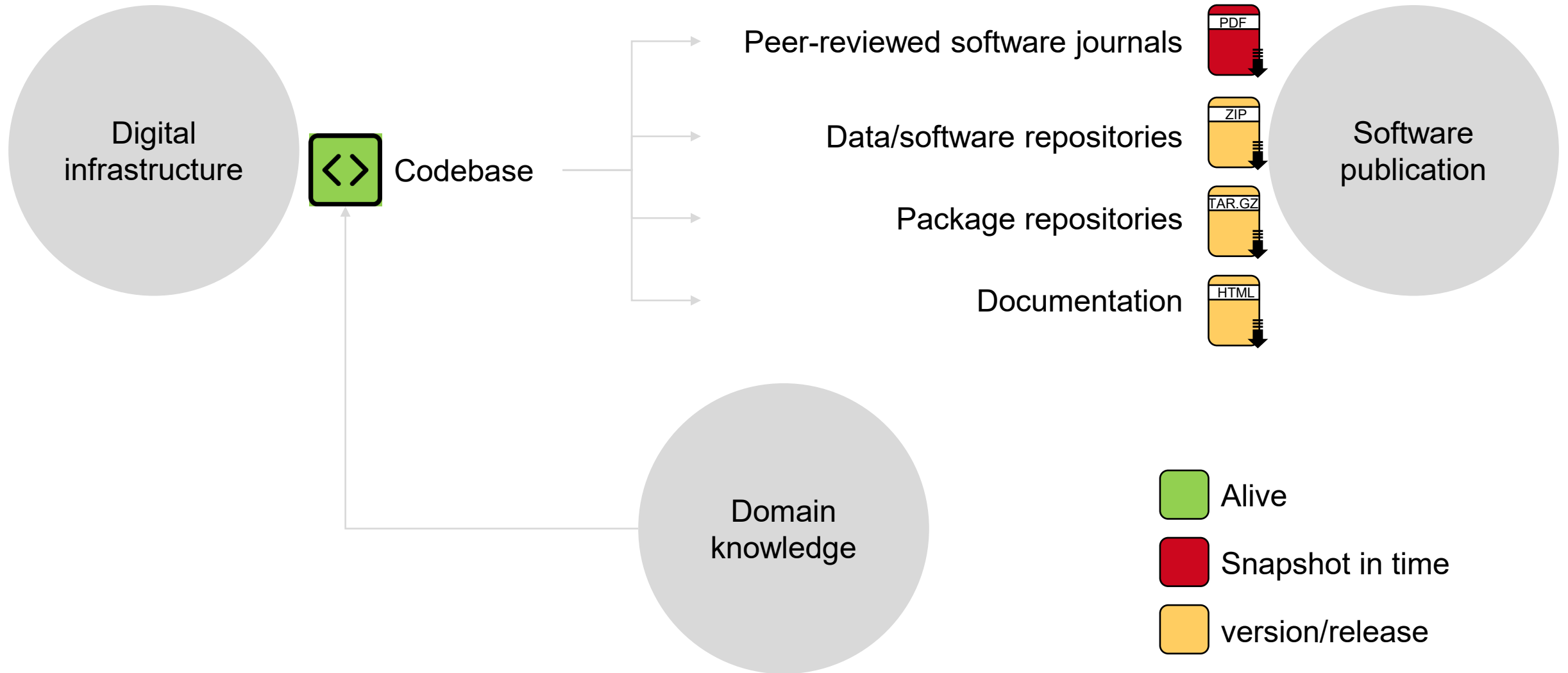
Pathways towards publishing



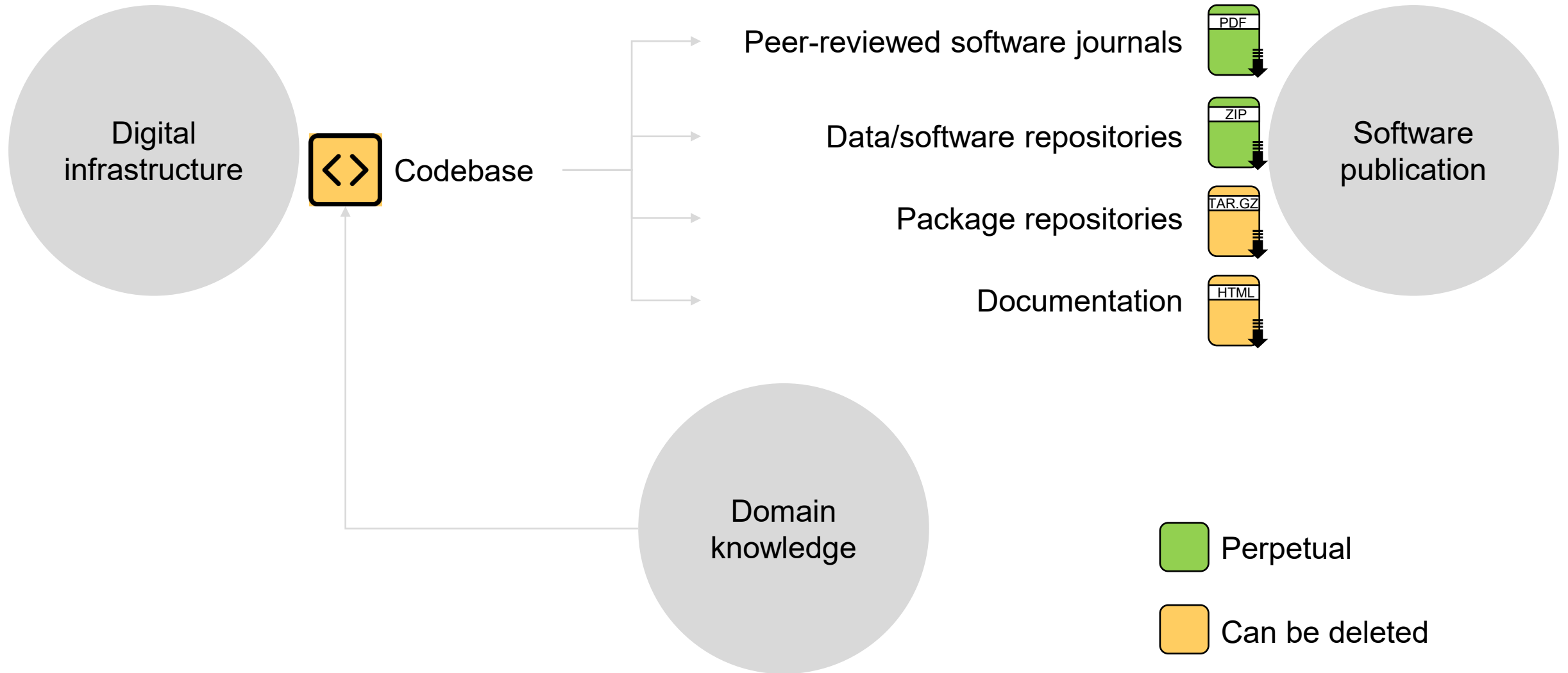
Pathways towards publishing



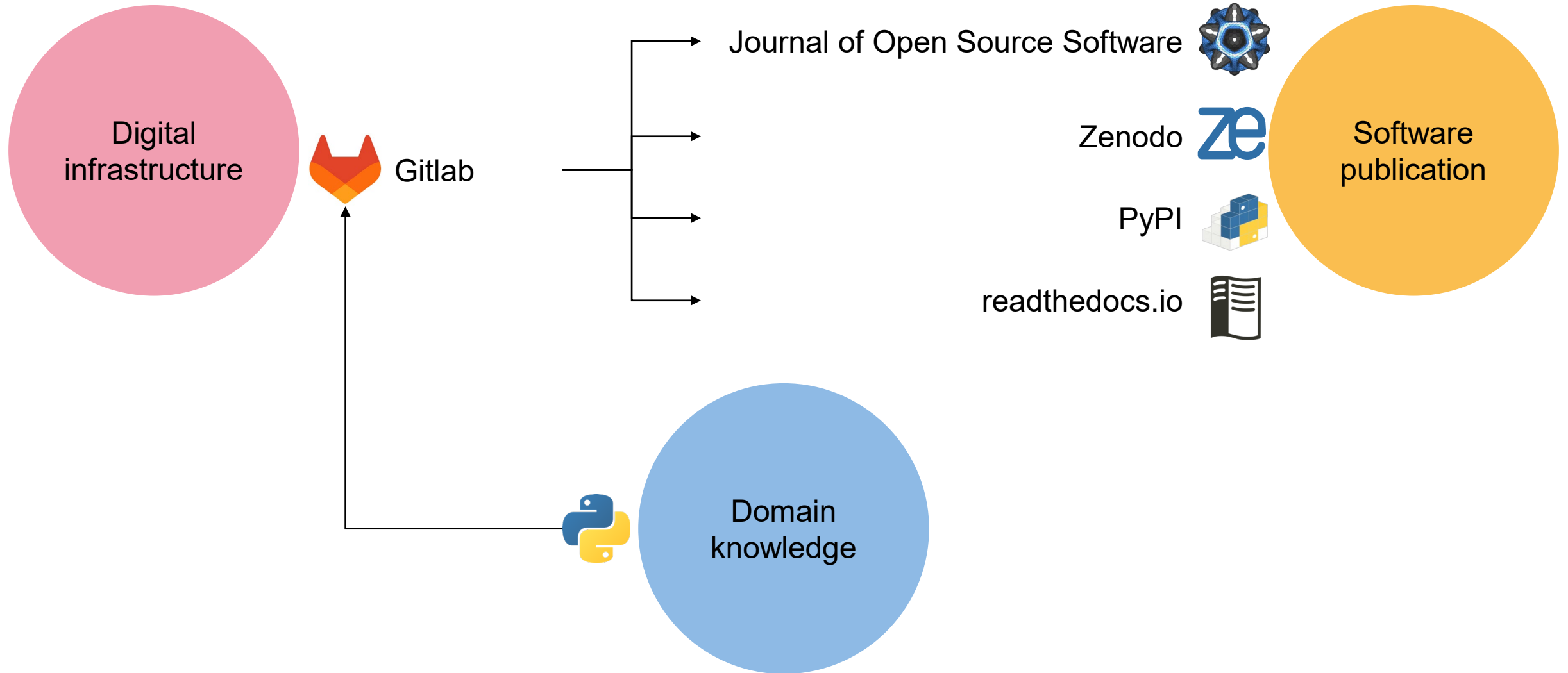
Pathways towards publishing



Pathways towards publishing



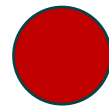
Pathways towards publishing – An example



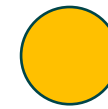
Final thoughts



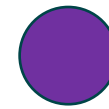
Institution



Chair / group



Project



Individual

Findability



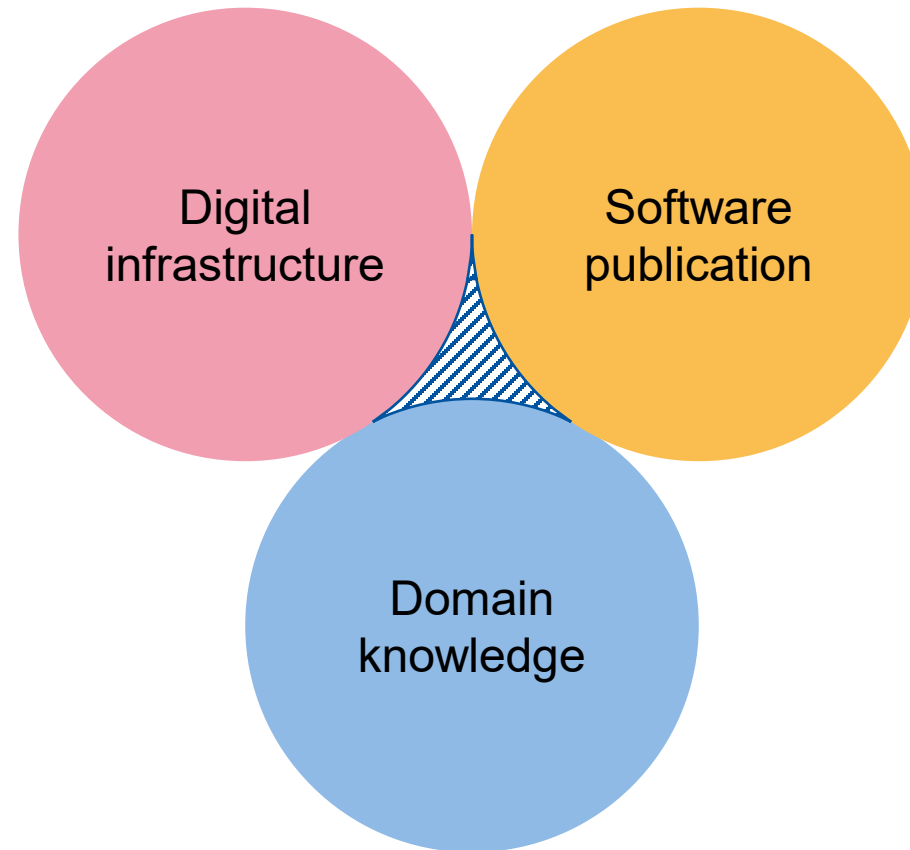
Responsibility of technical measures



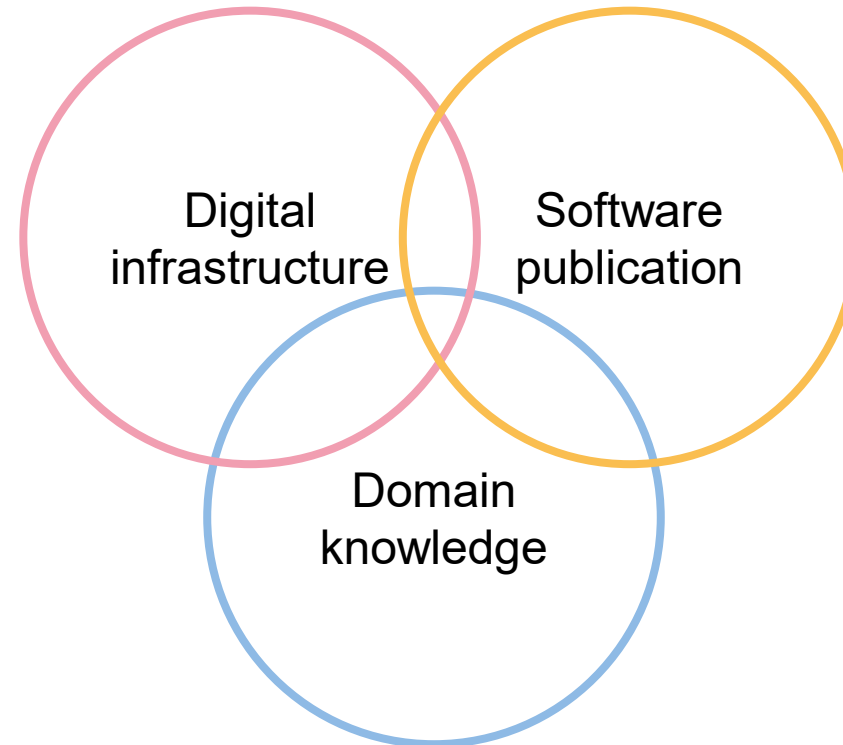
Responsibility of quality measures



Final thoughts



Final thoughts



*Great infrastructure does not unfold its value, if the software is bad.
Great software does not unfold its value, if there is no support.*

Thank you for your attention

interested in chatting with us about RSE?

rse@mbd.rwth-aachen.de

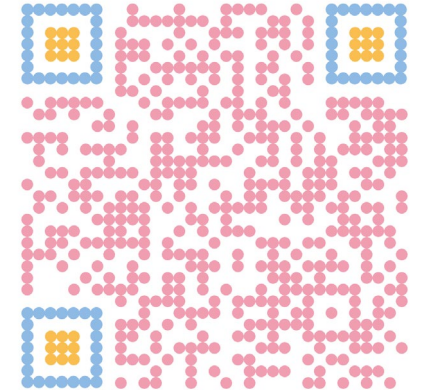
interested in our open-source software?

[mbd/software](#)

[github/mbd-rwth](#)

[gitlab/mbd](#)

[zenodo/mbd-rwth-aachen](#)





Research Software Engineering and Software Publication

RWTH Publishing Forum (#10) 25.9.2025

Dr. Dominik Schmitz, University Library, RWTH Aachen University

- Developing (research) software nowadays happens at platforms such as

GitHub



GitLab

or local installations such as

- <https://git.rwth-aachen.de>,
 - <https://git-ce.rwth-aachen.de>,
 - <https://git.nrw>, or
 - ... (your institution, partners, ...)
- But: **providing open access ≠ publication!**

Treat (research) software similar to any other research output
(such as paper or data): → Publishing and Citation

Why is „publishing“ different than „providing access“?

- If you develop your project publicly at a typical platform or if you switch it to public once you completed a state that you want to arrive at, everybody has access to this project and can use it, build on it.
- So it's *published*, right?

- Do you remember „SourceForge“?
- Do you remember when GitHub was bought by Microsoft?
- Do you know that GitLab @ RWTH has a „lifecycle“?
 - Of course you have many and simple means to ensure that *your* project is living for a long, long time in particular as long as it is developed actively.
- Do you know the version control portal to be used in 2050?
- You know that you can delete any project that you own, right?

What is so special about „publishing“?

- Somebody takes *responsibility* to make something available.
- And there is no agreed end of lifetime.
- It is the institutions or publishers duty to fulfill this responsibility for the unforeseen future through all organizational, administrative or technical evolutions.
- And even if an institution such as the RWTH Aachen University Library does not charge you, this does not mean there are no costs associated with it!
- But it explains why we need to boil it down to the most basic issues to ease and ensure future maintenance:
 - You get a persistent identifier, i.e. a PID, nowadays most often a DOI (= digital object identifier).
 - Basic metadata are associated with your research output – ready for dissemination and citation.
 - The PID resolves to the resource however and wherever this will be provided now or in the future.

→ Publishing and Citation ✓

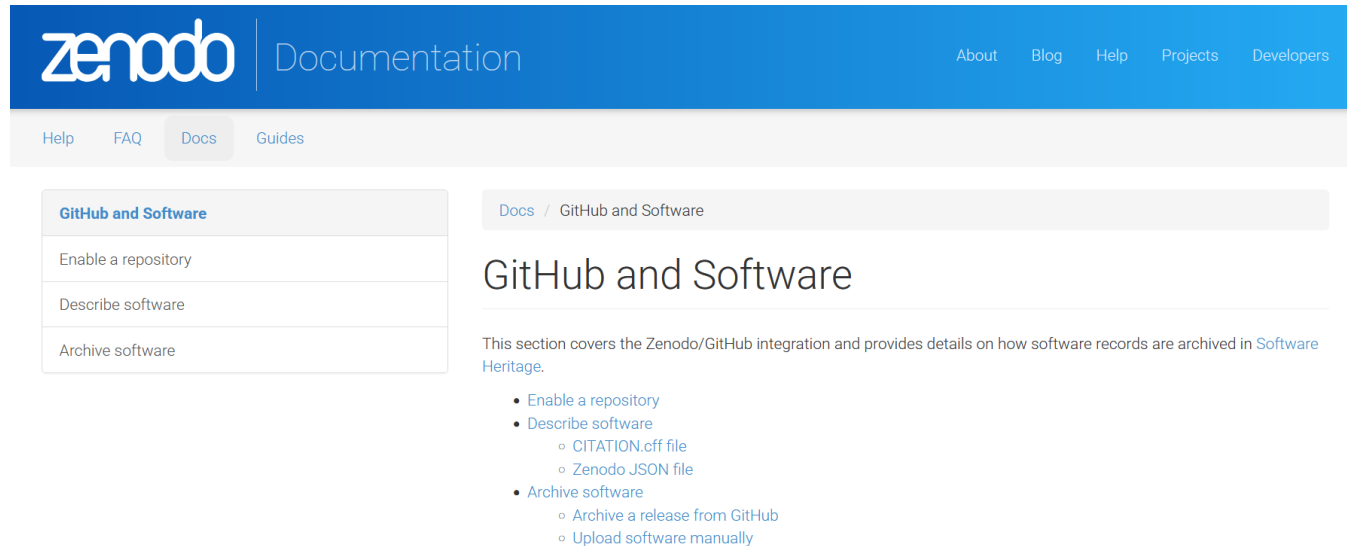
Publishing Software – How to do that nowadays?

- Publishing requires – just as with data or paper – a *snapshot* of your software, typically a *release*.
- The release is exported from wherever you develop your software (e.g. Git...) to a *repository*.
- You add suitable *metadata* in particular authors, a title, a year etc. (e.g. via a Citation File Format (.cff) file)
- You can keep a link to the version control source but it is just that – a link, i.e. just some other metadata.
The repository does not depend on the existence of the source!
- You get a *PID* and your software is citable.

- Concrete: GitHub projects?

→ [GitHub/Zenodo integration](#)

- You grant Zenodo access to your repo and it helps to gather the snapshot, provides a conceptual DOI, that always refers to your latest published release
- RWTH Publications: Currently manually, in the future similar approach for GitLab projects based on [HERMES](#) project.



The screenshot shows the Zenodo Documentation page for "GitHub and Software". The page has a blue header with the Zenodo logo and "Documentation" text. Navigation links include "About", "Blog", "Help", "Projects", and "Developers". Below the header, there are tabs for "Help", "FAQ", "Docs", and "Guides". The main content area is titled "GitHub and Software" and includes a sub-header "Docs / GitHub and Software". The page content states: "This section covers the Zenodo/GitHub integration and provides details on how software records are archived in Software Heritage." It lists three main topics: "Enable a repository", "Describe software" (with sub-items "CITATION.cff file" and "Zenodo JSON file"), and "Archive software" (with sub-items "Archive a release from GitHub" and "Upload software manually").

SWHID – SoftWare Hash Identifier

The software engineers will probably like this, but it gets a little bit technical...

<https://www.swhid.org/> <https://www.swhid.org/specification/v1.2/>

- ISO/IEC international standard 18670 on April 23, 2025
- „[...] persistent, *intrinsic* identifiers for software source code artifacts such as source code files, source trees, commits, and other objects typically found in version control systems”
- “pinpoint the exact version of any software artifact, at all levels of granularity, without relying on any central registry or naming authority”
- “can be computed using cryptographically strong functions directly from the digital objects they refer to, by anyone that has access to a copy of those objects. This enables decentralised and independent verification of integrity, without relying on a registry or a central authority.”

- Ursprung:
→ Software
Heritage
Archive

<https://archive.softwareheritage.org/>

Permalink – Resolving service
SWHID

swh:1:cnt:329ac324cd1338ad64be6a59899a1b9ada6864d0

;origin=https://git.rwth-aachen.de/guerrilla-analytics/project.git

;visit=swh:1:snp:ac0950d3c2cc8b80ca01fa4c3655743374d20da0

;anchor=swh:1:rev:d8368ff778cd064d06ed6b5bc0b65ee4e48dfced

;path=/README.md

Context qualifiers

Software Heritage Archive

<https://www.softwareheritage.org/>

Saving Research Software: Supporting Open Science

Home / Saving Research Software: Supporting Open Science

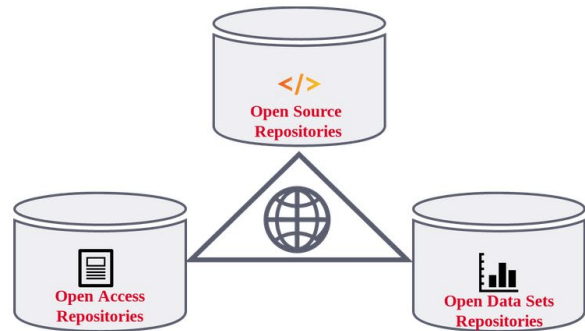
Looking for detailed guidelines? Go to the HOWTO archive and reference research software.

A pillar of Open Science

"Software should be considered a legitimate and citable product of research." — Software citation principles

"Non-reproducible single occurrences are of no significance to science."

— Karl Popper, *The Logic of Scientific Discovery*, 1934



Software has become a pillar of research, ubiquitous in all its fields: a large part of the technical and scientific knowledge being developed today is described in the software source code at a level of detail that is often needed to remove ambiguities that may

Supporting researchers in their quest for more reproducible research

To guide you on your journey, here are detailed guidelines on [how to archive and reference research software](#).



Software Heritage

- 1 Prepare your public repository
README, AUTHORS & LICENSE files
- 2 Save your code
<http://save.softwareheritage.org/>
- 3 Reference your work
(full repository, specific version or code fragment)

The Software Hash Identifier – SWHIDs for Academia

Building a solid web of knowledge that lasts over time is of paramount importance for academia. A key component of this is the

<https://www.softwareheritage.org/>

Saving Research Software: Supporting Open Science

Home / Saving Research Software: Supporting Open Science

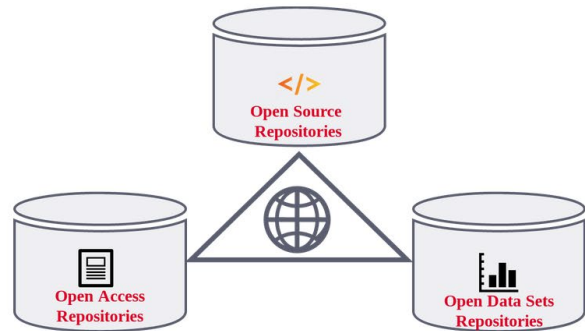
Looking for detailed guidelines? Go to the HOWTO archive and reference research software.

A pillar of Open Science

"Software should be considered a legitimate and citable product of research." — Software citation principles

"Non-reproducible single occurrences are of no significance to science."

— Karl Popper, *The Logic of Scientific Discovery*, 1934



Software has become a pillar of research, ubiquitous in all its fields: a large part of the technical and scientific knowledge developed today is described in the software source code at a level of detail that is often needed to remove ambiguities t

Supporting researchers in their quest for more reproducible research

To guide you on your journey, here are detailed guidelines on [how to archive and reference research software](#).

Why Software Attribution Is Challenging

Attributing credit in software projects is a complex issue, involving nuanced questions about who deserves credit, what they should be credited for, and how credit should be assigned. Here's a breakdown of the main challenges:

1. Who merits credit and for what?

- Software projects involve a wide range of roles beyond writing code, such as testing, documentation, design, maintenance, and project management. The traditional concept of "author" or "contributor" is too simplistic to capture these varied contributions.
- Inspired by systems like CRediT for research articles, frameworks such as Inria's study and the SORTÆD taxonomy propose detailed role classifications to reflect the complexity of software development.

2. Limitations of automated attribution tools

- Automated methods, like extracting author lists from git commit histories, are inadequate. They fail to recognize the diversity of contributions, focus disproportionately on coding, and overlook essential roles that do not appear in version control logs.

3. How should credit be given?

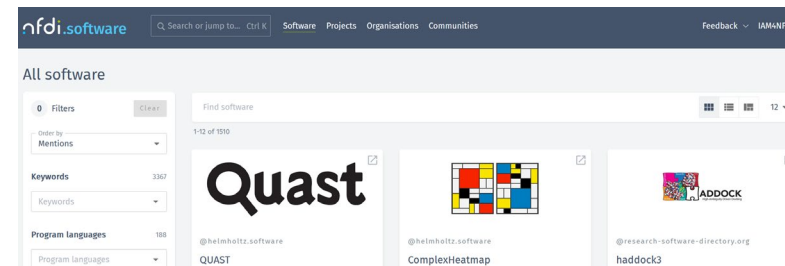
- In cases where the list of contributors is too long for practical citation, naming the project team (e.g., "The Givaro group") as the author is a common workaround.
- While this practice simplifies attribution, it raises questions about whether it sufficiently recognizes individual contributions or obscures the diversity of roles within a project.

Research Software Visibility

- Does RWTH need an organization at GitHub?
 - You should not be required to move your project to a new namespace.
 - A project might follow the researcher to other institutions.
 - And what about the projects at GitLab? Or at git.rwth-aachen.de or closed source?
- Recommendation for now
 - Tell us about your research software at [RWTH Publications](#)!
 - If Zenodo does not fit you, you can get a DOI for your most recent release here.
 - Thereby, DataCite will know about your software, BASE will, OpenAIRE will,...
 - But also: RWTH will know about it **and** will associate it correctly with you as a *person* and your *institute* probably even a *grant* (with your help)!
 - If you run a public project at a common platform you will highly likely be already archived by the Software Heritage Archive! If not make sure you are: <https://save.softwareheritage.org>
 - Use a SWHID if you need to refer in more detail.



- Base4NFDI Project <https://nfdi.software>
NFDI research software and data



Many thanks! Questions? Comments?

Dr. Anil Yildiz, Chair of Methods for Model-based Development in Computational Engineering, RWTH Aachen University, Aachen, Germany

E-Mail: yildiz@mbd.rwth-aachen.de

ORCID-ID: [0000-0002-2257-7025](https://orcid.org/0000-0002-2257-7025)

Dr. Dominik Schmitz, University Library, RWTH Aachen University, Aachen, Germany

E-Mail: d.schmitz@ub.rwth-aachen.de

ORCID-ID: [0000-0002-2362-4372](https://orcid.org/0000-0002-2362-4372)

DOI: [10.18154/RWTH-2025-07855](https://doi.org/10.18154/RWTH-2025-07855)



This work is licensed under [CC BY 4.0](https://creativecommons.org/licenses/by/4.0/). Excluded from the license are all logos, images and third party materials.