Research Software - Challenges, Pitfalls, Opportunities

22th October 2025 I Robert Speck | Jülich Supercomputing Centre at Forschungszentrum Jülich

“Domain scientists/HPC experts write great code!”

```c
#include <mpi.h>
#include <stdio.h>
#include <stdlib.h>
#define N 8
#define IDX(i,j) ((i)*N+(j))

int main(int argc,char**argv) {
    int rank, size, n, i, j;
    double *A=0, *x=0, *y=0, *localA, *localY;

    MPI_Init(&argc,&argv);
    MPI_Comm_rank(MPI_COMM_WORLD,&rank);
    MPI_Comm_size(MPI_COMM_WORLD,&size);

    if (N % size) {
        if (!rank) fprintf(stderr,"N not divisible by size\n");
        MPI_Abort(MPI_COMM_WORLD,1);
    }
    n = N/size;

    if (!rank) {
        A = malloc(N*N*sizeof(double));
        x = malloc(N*sizeof(double));
        y = malloc(N*sizeof(double));
        for (i=0;i<N;i++) {
            x[i] = 1.0;
            for (j=0;j<N;j++) A[IDX(i,j)] = i+j;
        }
    }

    localA = malloc(n*N*sizeof(double));
    localY = calloc(n,sizeof(double));

    MPI_Scatter(A,n*N,MPI_DOUBLE,localA,n*N,MPI_DOUBLE,0,MPI_COMM_WORLD);
```

Heard that one before? How about:

- 🗣️ “I’m an expert, my code is great!”
- 🗣️ “Meh, nobody looks at my code anyway..”
- 🗣️ “I (and only I) know my code best!”

Assumption: HPC users = code professionals, i.e.

- The code is readable and understandable
- The code is documented properly
- It comes with a readme and installation guides
- Examples show how to use it
- It has **tests to show that it works** properly
- It is citable
- You name it…

Observation: often not true

When code goes rogue…

illegal Conversion between data types caused Arianne 5 rocket to explode 430 seconds after lift off.

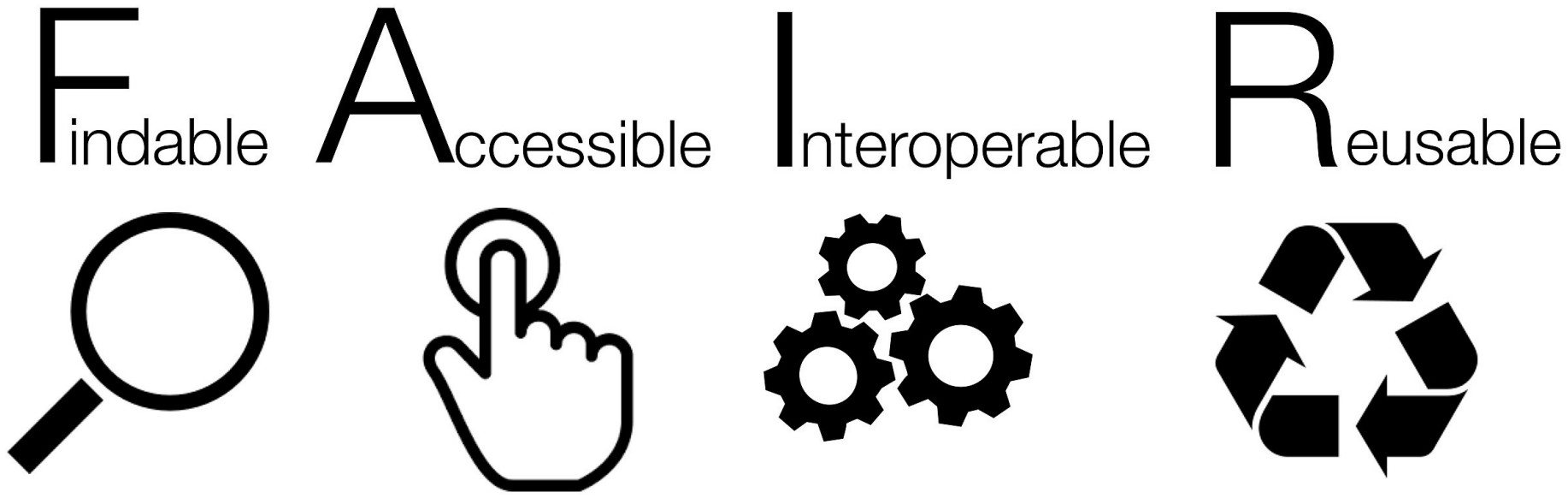
Science

A Scientist's Nightmare: Software Problem Leads to Five Retractions

Has a software bug really called decades of brain imaging research into question?

The Guardian, 2016

illegal mixing of different units of measurement (SI, Imperial, and US) caused the Mars Climate Orbiter to be lost on entering orbit around Mars in 1999.



Part I: Introduction to Research Software Engineering

Robert Speck | Jülich Supercomputing Centre at Forschungszentrum Jülich GmbH | 2025

Are you working with research software? (Spoiler: yes)

A strict/exclusive definition of research software

- Well identified software that is part of the research discovery process, which might require specialized domain knowledge and is by itself a contribution to science and research
- Software that was developed with the intention of being part of research

Examples of research software

- An application software, serial, parallel, GPU, CPU, big machine, small machine, …
- A software other people use for their scientific work: a benchmarking tool, a profiler, a plotting script, …


FORCE11 RDA RESEARCH DATA ALLIANCE <ReSA> Research Software Alliance

Defining Research Software: a controversial discussion

Summary Report of FAIR4RS Subgroup 3 activity and discussion

Morane Gruenpeter (Inria, Software Heritage), Daniel S. Katz (University of Illinois), Anna-Lena Lamprecht (Utrecht University), Tom Honeyman (Australian Research Data Commons), Daniel Garijo (Information Sciences Institute), Alexander Struck (Cluster of Excellence Matters of Activity, de-RSE), Anna Niehues (Radboud university medical center), Paula Andrea Martinez (Research Software Alliance), Leyla Jael Castro (ZB MED Information Centre for Life Sciences), Tovo Rabemanantsoa (French National Research Institute for Agriculture, Food and Environment), Esther Plomp (Delft University of Technology - Faculty of Applied Sciences), Neil Chue Hong (Software Sustainability Institute / EPCC, University of Edinburgh), Carlos Martinez-Ortiz (Netherlands eScience Center), Laurents Sesink (Leiden University Libraries), Matthias Liffers (Australian Research Data Commons), Anne Claire Fouilloux (University of Oslo, Department of Geosciences), Chris Erdmann (American Geophysical Union), Silvio Peroni (University of Bologna), Paula Martinez Lavanchy (TU Delft Library), Ilian Todorov (UKRI STFC), and Manodeep Sinha (Swinburne University of Technology)

https://zenodo.org/records/5504016

Examples of non-research software (in the exclusive sense)

- Software used for science, but not developed for it (Python, Excel, …)
- Software used in the scientific workflow, but not for scientific discovery (e.g., Linux)

⚠ Did I mention Excel? Excel is NOT a research software.

Difference between research and industry software

Industry software

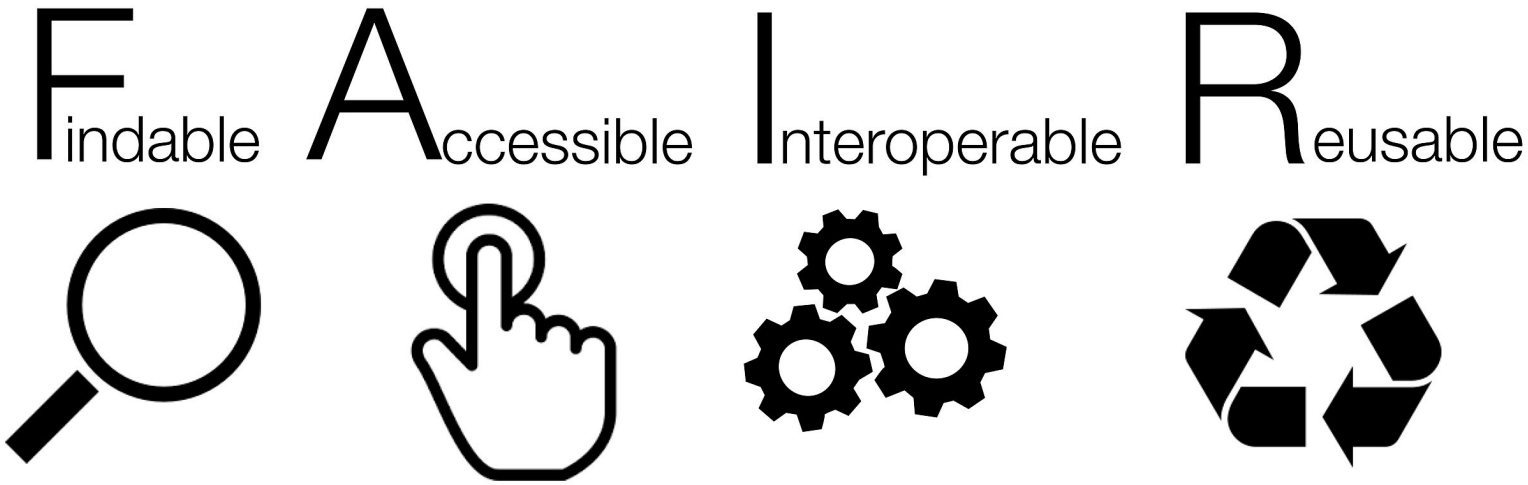
- Result/behavior usually clearly defined
- Should follow a **clear development plan**
- Significant impact on budgets
- Mostly written by **specialized software engineers**
- Usually a clearly defined target
- **Profit-oriented** with a target market
- Should follow good software engineering practices

Research software

- Has usually **unknown results**
- Can evolve from a PoC to a full application
- Usually **not part of any budget**
- Rarely written by specialized software engineers
- Changing target, **changing requirements**
- Usually not profit-oriented
- Should follow **good software engineering practices**

Important aspects of RSE

The FAIR4RS way



From:
Barker, M., Chue Hong, N.P., Katz, D.S. et al. **Introducing the FAIR Principles for research software**. Sci Data 9, 622 (2022). https://doi.org/10.1038/s41597-022-01710-x

F: Software, and its associated metadata, is easy for both humans and machines to find.
A: Software, and its metadata, is retrievable via standardized protocols.
I: Software interoperates with other software […] via APIs, described through standards.
R: Software is both usable and reusable.

FAIR4RS: application of the FAIR principles to research software

⚠ Note: research software is NOT (only) data!

From FAIR to RSE

Making/keeping research software FAIR

We can use the **FAIR4RS principles** to derive good practices for research software:

- Documentation — from: **I**nteroperable, **R**eusable
- Community — from: **A**ccessible, **F**indable
- Versioning — from: **F**indable, **A**ccessible
- Automation — from: **I**nteroperable, **R**eusable

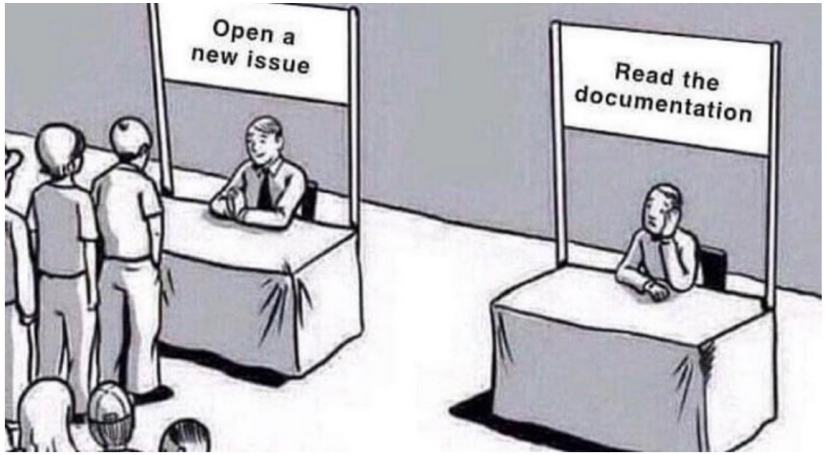
Note: There are many more approaches to “define” what RSE comprises, e.g.:

- Best Practices for Scientific Computing, Wilson et al., 2014, PLOS
- Good Enough Practices in Scientific Computing, Wilson et al., 2017, PLOS
- de-RSE position paper and resource collection on RSE competencies
- Four simple recommendations to encourage best practices in research software, Jimenez et al., 2017

+ an unknown number of other publications, websites, opinions, …

⚠ Bottom line: there is no single standard definition or curriculum of RSE techniques!

FAIR4RS in practice: Documentation

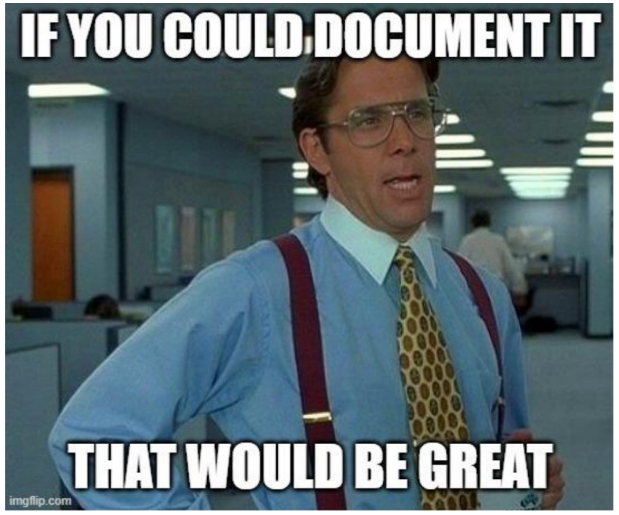


What?

- Describe what the code is doing and how to install it
- Have a license that comes with your code

Why?

- Users want to understand how to work with your code
- Also, Future You will thank Present You for this



https://zenodo.org/records/7260347

How?

- Use established tools/conventions, do not re-invent the wheel
- Try to be consistent and consequent

FAIR4RS in practice: Community

What?

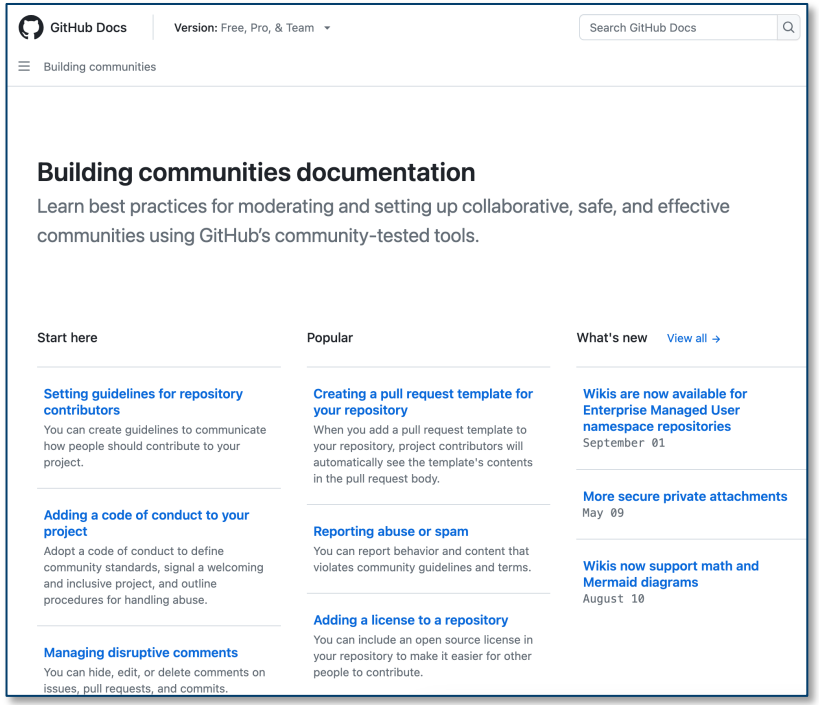
- Easy: Have issue tracker, contribution guidelines
- Pro: Run a mailing list, meet people/present at conferences

Why?

- Users should know how to get in touch and contribute
- Being open for contributions is the only way to get them

How?

- Follow established standards/guidelines (e.g., at GitHub)
- “Treat others how you want to be treated”



https://docs.github.com/en/communities

FAIR4RS in practice: Versioning

What?

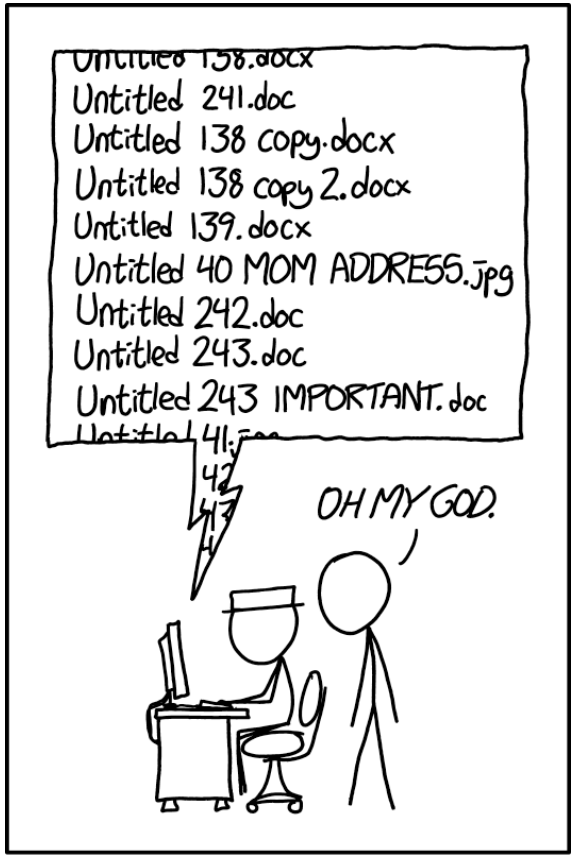
- Make sure your code is version controlled
- Each version should have a unique identifier

Why?

- Users need to know what version they work with
- Future You will thank Present You once you track changes

How?

- Use a service like gitlab or GitHub, gives you commit hashes
- Write clear commit messages and release notes



PROTIP: NEVER LOOK IN SOMEONE ELSE'S DOCUMENTS FOLDER.

https://xkcd.com/1459/

FAIR4RS in practice: Automation

What?

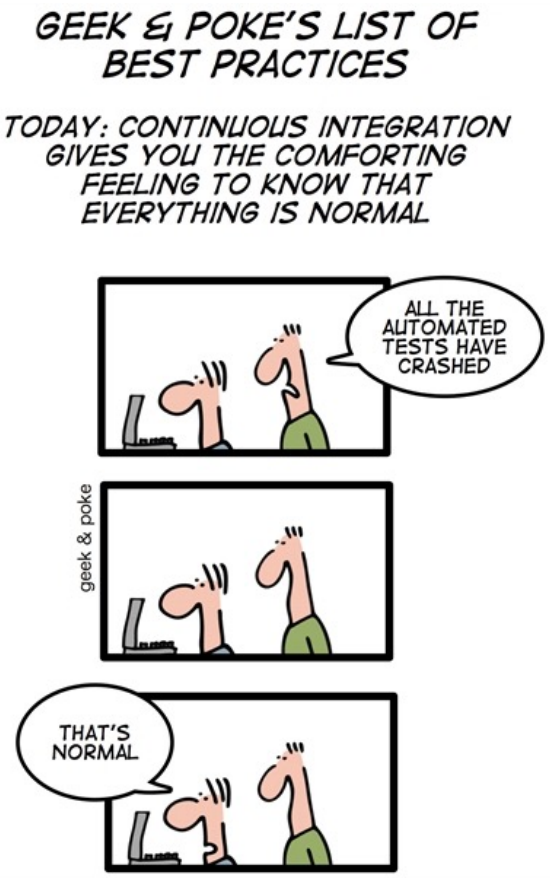
- Have tests for your code showing correctness and usability
- Have an automated way to run these tests

Why?

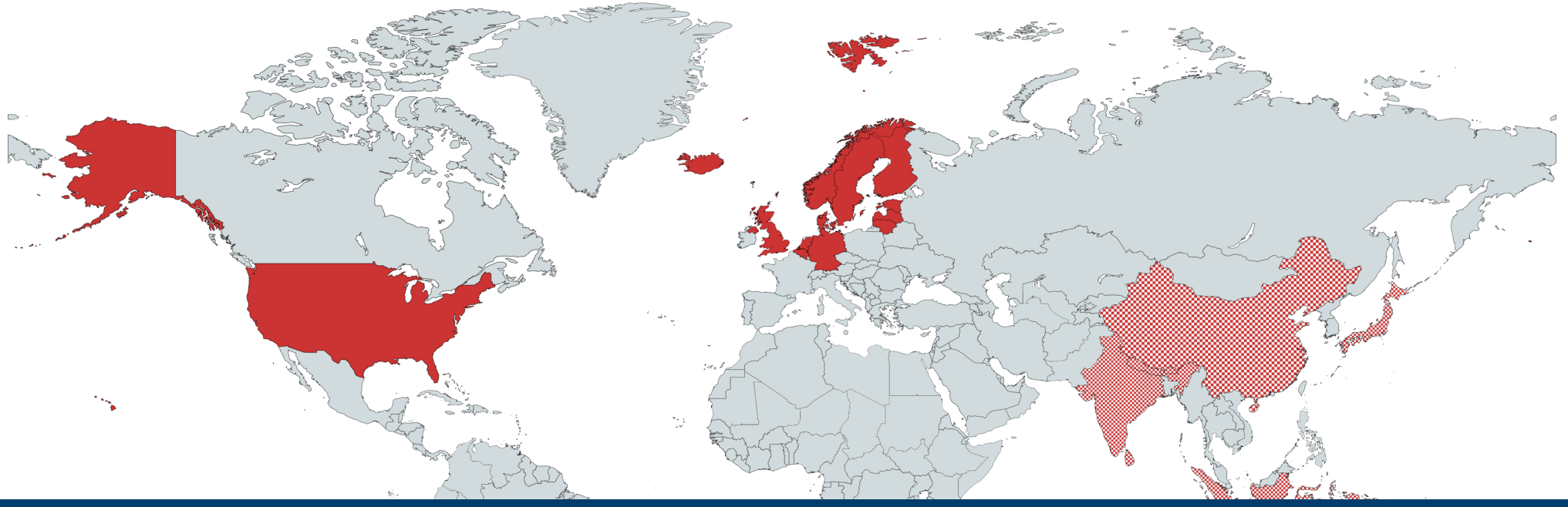
- Tests strengthen the trust in your work, for users and yourself
- Automation reduces the human factor

How?

- Use continuous integration pipelines offered e.g. by GitHub
- Follow established standards, do not re-invent the wheel



https://geek-and-poke.com

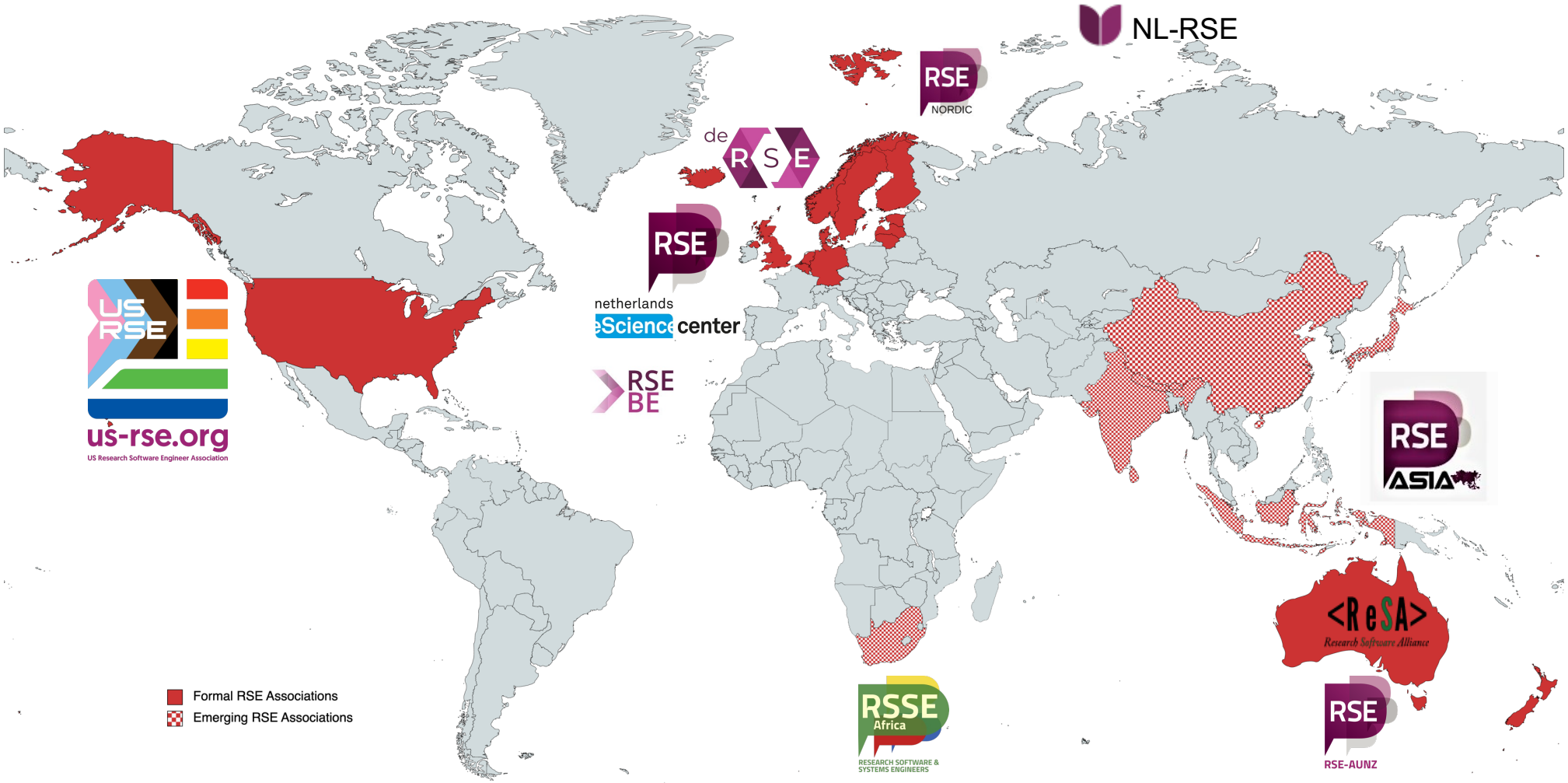


Part II: RSE Landscapes

Robert Speck | Jülich Supercomputing Centre at Forschungszentrum Jülich GmbH | 2025

The worldwide RSE movement

intl RSE | INTERNATIONAL COUNCIL OF RSE ASSOCIATIONS



Code for thought
RSE

JÜLICH Forschungszentrum

Germany's RSE landscape

Nation-wide:

- de-RSE e.V.
- FutuRSI (see later)

Domain-specific:

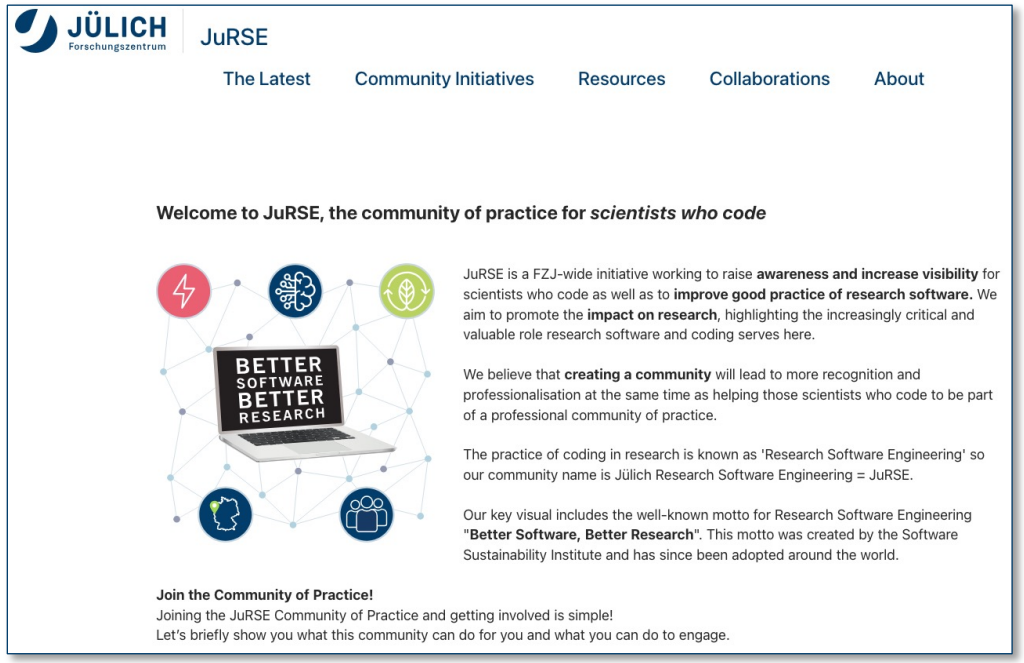
- Gesellschaft für Informatik: Fachgruppe RSE
- Gesellschaft für Mathematik und Mechanik: Fachausschuss RSE
- NFDI: nfdi.software inside Base4NFDI (and more)

State-level:

- "Rahmenkonzept der Universitäten des Landes Baden-Württemberg für das High-Performance Computing und Data-Intensive Computing"
- "Lower Saxony Digital Science Support Space"

+ lots of **regional/local groups** with varying visibility, impact, and outreach

JuRSE – Jülich RSE Community of Practice



https://go.fzj.de/JuRSE

- **Increase good practice, but also visibility and awareness**
- **Encourage adoption of the FZJ software and publication guidelines**

JuRSE – Community and visibility activities

Join the community platform (online)

Meet other RSEs at FZJ and Germany!

Join us at JuRSE Open Hours

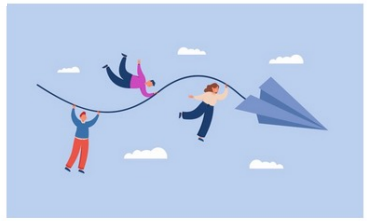
In-person and online, we're here to help!

JuRSE Travel Grants

Go to an RSE Conference on us!

JuRSE Code of the Month

A code in the spotlight



JuRSE Newsletters

Read about RSE News, events, blogs and podcasts

- **Increase good practice, but also visibility and awareness**
- **Encourage adoption of the FZJ software and publication guidelines**

Joint Lab HiRSE

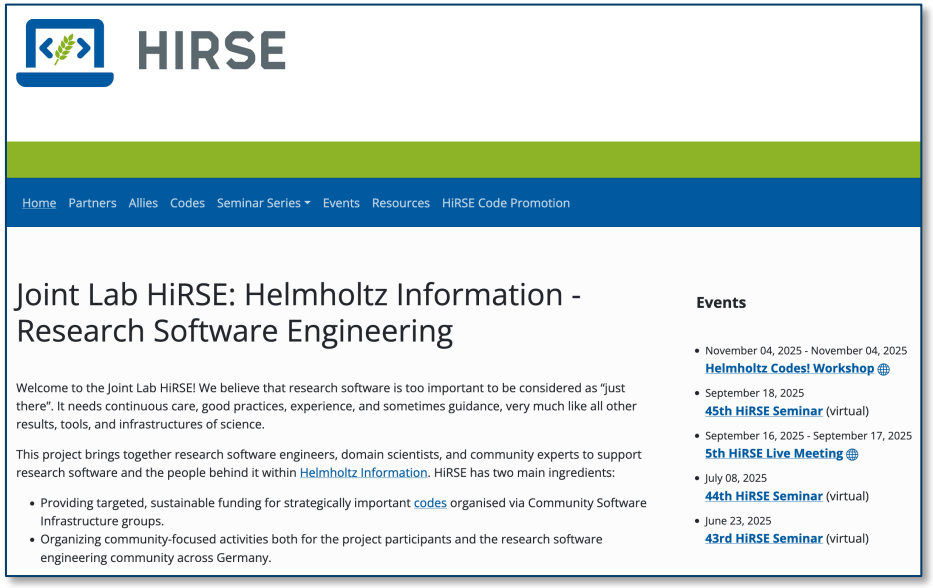
From a small-scale preparatory study to a wider vision

A vision

- Sustained funding for research software as infrastructure
- Dedicated embedded RSEs for long-term tasks (maintenance, documentation, porting)
- Trial run with 5 groups within Helmholtz

Services for the national RSE community

- HiRSE Seminar Series, 1-2 per month, with YouTube channel and Zenodo community
- HiRSE Summer of Testing and Summer of Programming Languages + Hackathons + Summer School
- HiRSE Code Promotion: research software made in Germany
- Contributions to national events and initiatives



https://www.helmholtz-hirse.de

The FutuRSI project

FutuRSi
Next-Level RSE in Germany

Thinking about a German Research Software Institute/Institution

A challenge

- Does Germany need a central research software “something”?
- How could such a ”something” looks like, work, get funding?
- Examples: UK SSI, NL eScience Center, (US RSE), all with dedicated funding

Preparatory steps

- Creation of a feasibility study and a sound concept for a nationwide research software institution
- Documentation of the requirements, processes and steps for the prototypical development of such an organization.
- Advocacy for open, collaborative and sustainable practices in Research Software Engineering
- Provide a bigger picture of RSE in Germany, more here: https://www.futursi.de

Klaus Tschira Stiftung

RSE funding landscape

💰💰 **DFG**: dedicated RSE calls
💰💰 **DFG**: research software infrastructure funding

💰 **BMFTR**: nothing (yet)
💰 **BMDS**: nothing

💰💰💰 **EU**: EuroHPC lighthouse codes and other CoEs
💰💰 **EU**: EOSC

💰💰 Trusts: Klaus Tschira, Eclipse foundation, …

Missing pieces:

- BMFTR and NRW initiatives
- Sustained funding for RSEs and research software

Sources:

- Research Software Alliance Newsletter
- https://github.com/DE-RSE/funding_calls/issues
- de-RSE Matrix channel

⚠️ Most research software in Germany is either only indirectly funded or not at all!

Research software KPIs

Make research software count!

HELMHOLTZ
Open Science

What we want:

Make research software count as valuable output of science
Recognize and fund the effort it takes to write good software for science

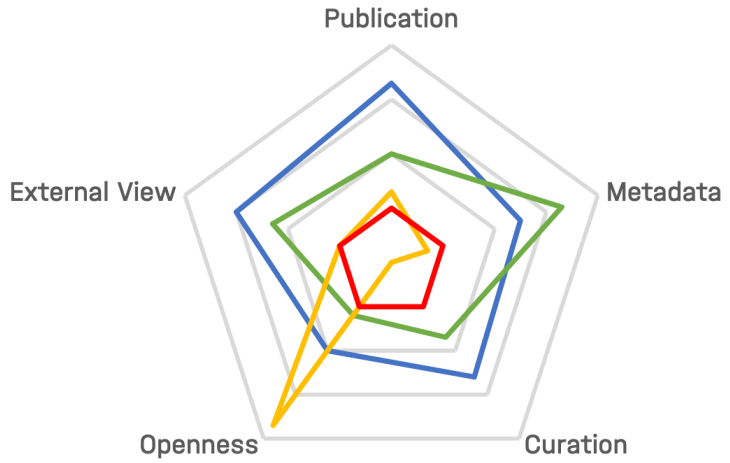
What we need (besides 💰):

Good ways to check the quality of research software (like we do for papers)
Good ways to actually count research software output (similarly for data)

Towards a Quality Indicator for Research Data publications and Research Software publications – A vision from the Helmholtz Association

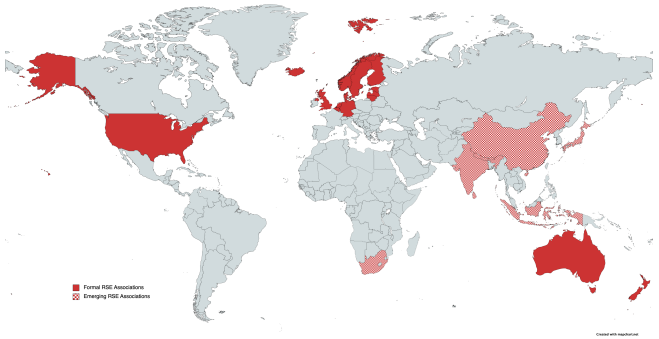
Wolfgang zu Castell[1], Doris Dransch[1], Guido Juckeland[2,*], Marcel Meistring[3], Bernadette Fritzsch[4], Ronny Gey[5], Britta Höpfner[6], Martin Köhler[7], Christian Meeßen[1], Hela Mehrtens[8], Felix Mühlbauer[1], Sirko Schindler[9], Thomas Schnicke[5] and Roland Bertelmann[3]

https://doi.org/10.48550/arXiv.2401.08804



JÜLICH
Forschungszentrum

Three take aways

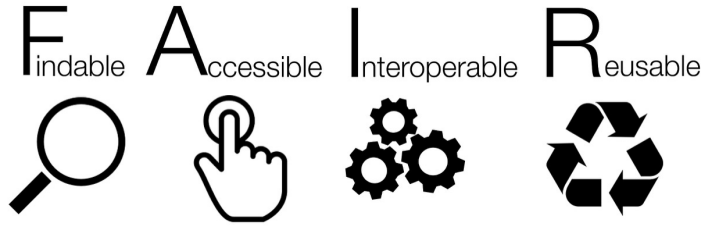


Research Software Engineering worldwide

- Research software is a valuable output of science
- There is an active community in Germany and worldwide
- Lots of ways to support and receive support through this community

FAIR for Research Software

- There is no single definition of what research software engineering is
- Here: FAIR4RS as guideline, deriving methods from there
- Research software engineers are usually not software engineers



Bundesministerium für Forschung, Technologie und Raumfahrt

DFG Deutsche Forschungsgemeinschaft

EuroHPC Joint Undertaking

Sustainable research software, sustainable funding

- Current research software funding options are rare
- Key performance indicators/peer review procedures are a big issue
- Some (promising) initiative try to change that