deRSE24 - Conference for Research Software Engineering in Germany

Tuesday, 5 March 2024 - Thursday, 7 March 2024

Julius-Maximilians-Universität Würzburg

Book of Abstracts

Contents

Continuous Integration - Basic / **1**

Getting up and Running with your first CI Pipeline

Author: Maximilian Beuscher[None]

Co-authors: Florian Goth [1]; Yanick Thurn [2]

[1] *Universität Würzburg*

[2] *Julius-Maximilians Universität Würzburg*

Corresponding Author: florian.goth@physik.uni-wuerzburg.de

This is a tutorial aimed at absolute beginners that tries to get people started with their first CI pipeline. This is a skill-up session. We will use gitlab try to get a linter working on a python script. The workshop will utilize a carpentries style such that everyone leaves with a working CI example. This workshop is part of the “A day of CI” track.

Slot length:

Teaching RSE Skills / **2**

State of the TeachingRSE Project: Foundational competencies of an RSE and what comes next

Author: Florian Goth[1]

[1] *Universität Würzburg*

Corresponding Author: florian.goth@physik.uni-wuerzburg.de

At a community workshop at deRSE23 in Paderborn we talked about
teaching RSEs. Among the questions considered were, what are the basic competencies of RSEs?
Which institutions are required? Which changes are necessary in the current academic system?
After this workshop we started to gather the input from the community
and quickly noticed that the scope is bigger than a single short publication,
and therefore the TeachingRSE project was born.
In this talk I will present our first publication, on the foundational competencies of an RSE, arXiv:2311.11457
. In addition I will detail the direction of this project and point people to possibilities for contributing
online, or directly at the accompanying workshop.

Slot length:

Workflowmanagement for Parallel Computing / **4**

Taskfarm: A Client/Server Framework for Supporting Massive Embarrassingly Parallel Workloads

Author: Magnus Karl Moritz Hagdorn[1]

[1] *Charité Universitätsmedizin Berlin*

Corresponding Author: magnus.hagdorn@charite.de

Task farms can be used to solve embarrassingly parallel workloads where a number of independent tasks need to be performed. This presentation introduces *taskfarm*, a python client/server framework that was designed to manage a satellite data processing workflow with hundreds of thousands of tasks with variable compute costs. The server uses flask to hand out tasks via a REST API and a database to track the progress of tasks. The client is also implemented in python. The presentation will focus on the software design process, the pitfalls and dead ends encountered when dealing with big data and how they were resolved.

Slot length:

Poster Session / **6**

Using RSE skills and Open Source Software to propose a heterogenious Management Hub in NFDI4Objects

Authors: Fabian Fricke[1]; Florian Thiery[2]

[1] *Deutsches Archäologisches Institut*

[2] *Leibniz-Zentrum für Archäologie (LEIZA)*

Corresponding Authors: florian.thiery@leiza.de, fabian.fricke@dainst.de

NFDI4Objects (N4O) represents a broad community dealing with material remains of human history from around 3 million years. RSE skills and the combination of FOSS help to create a Management Hub. The hub of our planned management tool will be a Nextcloud instance. This instance will provide a set of basic services (file sharing, task management, notes, shared calendars) and other FOSS services can be easily integrated. These services include Rocket Chat for communication, OpenProject for project launch planning, Zammad as a helpdesk ticketing system, Collabora for online document editing, and LimeSurvey combined with ShinyR for surveys. On top of that, reports could be imported/exported into a Wikibase using the included frameworks to create a Management Knowledge Graph.

Slot length:

7

Meet-Up: Building a network of Research Software Engineers in the Leibniz Association

Authors: Florian Thiery[1]; Jan Philipp Dietrich[2]; Jan Philipp Thiele[3]

[1] *Leibniz-Zentrum für Archäologie (LEIZA)*

[2] *Potsdam Institute for Climate Impact Research (PIK)*

[3] *Weierstrass Institute Berlin*

Corresponding Authors: florian.thiery@leiza.de, thiele@wias-berlin.de, dietrich@pik-potsdam.de

In the Leibniz Association there are at least three institutes (e g., LEIZA, WIAS, PIK, ...) with active members in the RSE community - but we have the feeling that we might be more. For that reason we would like to come together in Würzburg to get to know each other, share and discuss ideas on research software engineering within the Leibniz family and build up a Leibniz-RSE network.

Options that we might want to discuss are the formation of a “deRSE Arbeitskreis RSE @ Leibniz” or some linking into the currently forming Working Group (AG) for Software in the Leibniz Association (in association with the WG Research Data).

How to participate and do networking as members of both Leibniz and the RSE community could also be a point of the meet-up discussions. Are there perhaps topics/fields where we should bring an RSE perspective into Leibniz or an Leibniz perspective into RSE? To support the discussions we will share both experience and appropriately branded cookies (even if we are not accociated with that specific company in Hannover).

Slot length:

other(help with comment)

Teaching RSE Skills / 8

Research software engineering - how do we teach the necessary skills to interdisciplinary teams?

Author: Astrid Nieße[1]

Co-author: Stephan Ferenz [1]

[1] *Carl von Ossietzky Universität Oldenburg*

Corresponding Author: astrid.niesse@uni-oldenburg.de

Research projects in the field of energy systems research are typically conducted in interdisciplinary teams - scientists from energy and electrical engineering, computer science, economics and other disciplines often work together on simulation-based analysis [Ferenz2022]. In many projects, joint simulation models are created and integrated into existing simulation frameworks. Analysis of data from simulation studies is also often undertaken as a joint activity. Thus, the need for joint software and model development in energy system research is obvious. Nevertheless, the knowledge of software engineering strongly varies between the different backgrounds. In such an interdisciplinary research project in the field of energy research, we tested whether a training series on research software engineering would improve the quality of the work, especially the software, and the mode of collaboration.

The training series served to harmonize the level of knowledge on research software engineering across different disciplines and to make the individual competencies and knowledge of the partners usable within the overall group. This is done with the clear goal of improving the quality of the work and making tools relevant to the work known to all.
As part of the training series, several topics have been discussed beforehand with research group leaders in the domain to identify relevant competencies in the given setting of energy system research software.
The topics of joint software development, version management, and (automated) testing of the developed software have been identified as crucial aspects from the DevOps area, and thus have been addressed in first trainings. This was undertaken to ensure that all partners have a consistent level of knowledge and can guarantee a consistent quality standard for the developed software. As a basic module, some aspects from requirements engineering have been been part of the curriculum.
In addition, a closer look at open source strategies and licensing models was included since these are increasingly important in the context of Open Science. In this way, it was ensured that project members can meet the requirements of this approach during their work.

In the talk, some lessons learned from conducting the course will be presented and discussed. We would like to get feedback on the general idea, the course concept, the curriculum and discuss experiences of other attendees on delivering this kind of knowledge to interdisciplinary research teams.

The talk should include 15 minutes + 15 minutes discussion with the audience, if possible.

[Ferenz2022] Ferenz, S.; Ofenloch, A.; Penaherrera Vaca, F.; Wagner, H.; Werth, O.; Breitner, M.H.; Engel, B.; Lehnhoff, S.; Nieße, A. An Open Digital Platform to Support Interdisciplinary Energy Re-

search and Practice—Conceptualization. Energies 2022, 15, 6417. https://doi.org/10.3390/en15176417

Slot length:

9

How to write a parser for DSLs and other purposes

Author: Eckhart Arnold[1]

[1] *Bayerische Akademie der Wissenschaften*

Corresponding Author: eckhart.arnold@badw.de

This is a hands-on tutorial for how to write a parser with parsing-expression-grammars (PEG). PEG-based parsers are essential for building domain-specific-languages. They are also useful for programming compilers and transpilers or for retrieving structured data from hand-written sources like bibliographies.

Writing formal grammars is often considered difficult. The key to success is, in my opinion, to employ an incremental test-driven approach. In this totorial we will use the DHParser-framework which is a Python-based Parser-generator with strong support for test-driven development.

As guiding example we will write a grammar for Markdown and construct a Markdown-parser from scratch. Along the way we will also touch such topics as simplification of syntax-trees, locating and reporting of syntax errors, fail-tolerant parsing.

If time permits we might in the afternoon session also cast a glance at more advanced topics like the use of macros and preprocessors or, depending on the preferences of the audience at more complicated examples like parsing LaTeX or transpiling data-structure-definition from Typescript to Python.

Most of the tutorial will be based on the documentation of DHParser on https://dhparser.readthedocs.io.

Prerequisits are a good knowledge of regular expressions (we will use them a lot as the modes building blocks for our grammar). People who want to follow through the examples and exercises themselves should bring a Laptop with Python 3.7 (https://python.org) or higher and PyCharm-Community-IDE (https://www.jetbrains.com/de-de/pycharm/) installed. Also DHParser should be installed with the command "python -m pip install DHParser" or directly from https://gitlab.lrz.de/badw-it/DHParser.

Slot length:

Workshop (3h)

RSE in Digital Humanities / 10

A domain-specific-notation for the Medieval-Latin-Dictionary

Author: Eckhart Arnold[1]

[1] *Bayerische Akademie der Wissenschaften*

Corresponding Author: eckhart.arnold@badw.de

The Medieval-Latin-Dictionary (MLW - https://mlw.badw.de) is a research project of the Bavarian Academy of Sciences and Humanities. It's purpose is to write comprehensive a dictionary of medieval latin taking into account scholarly (theological) as well a profane sources.

Writing Dictionaries of this kind and extend typically is an enterprise that takes many decades to complete. And the Medieval-Latin-Dictionary, too, started decades ago. Several years ago, its workflows have been put on a fully digital basis, which also includes entering the dictionary articles in a structured form.

For this purpose often XML is the technology of choice. However, XML is clumsy to read and write and, because of that, one can hardly avoid to use proprietory technolgie like the Oxygen-XML-Editor for authoring. The scientists had been given the choice to either use XML or try a domain specific notation which was going to be developed specifically for the MLW.

After prototypes for both alternatives had been presented, the MLW-team soon decided in favor of using a domain-specific-notation which is now in use for three years.

In my talk I describe the collaborative process in which the DSL was developed and launched, its key-design-principles and technological choices. However, I will also talk about the things that went wrong and the lessons we learned.

(The software as well as the documentation is Open Source and available on: https://gitlab.lrz.de/badw-it/mlw-dsl-oeffentlich)

Slot length:

other(help with comment)

Poster Session / **11**

Composable Bayesian Inference with BlackJAX

Authors: Adrien Corenflos[1]; Alberto Cabezas[2]; Junpeng Lao[3]; Rémi Louf[None]

[1] *Aalto University*

[2] *Lancaster University*

[3] *Google*

Corresponding Author: alb.cab94@gmail.com

BlackJAX is a library implementing sampling and variational inference algorithms commonly used in Bayesian computation. It is designed for ease of use, speed, and modularity by taking a functional approach to the algorithm's implementation. Designed from basic components to specific iterative procedures, BlackJAX allows the end user to build and experiment with new algorithms by composition. BlackJAX is written in pure Python using JAX to compile and run NumpPy-like programs on CPUs, GPUs and TPUs. The library integrates well with probabilistic programming languages by working directly with the (unnormalized) target log density function, given that the function is pure. The library is intended for users who need to create complex sampling mechanisms beyond the black-box solution, researchers who want to experiment when developing new algorithms and students who want to learn how inference algorithms work.

Slot length:

RSE Practices / **12**

flowR: A Program Slicer for the R Programming Language

Author: Florian Sihler[1]

Co-author: Matthias Tichy [1]

[1] *Ulm University*

Corresponding Author: florian.sihler@uni-ulm.de

Context:
Program slicing 1 is an important technique to assist program comprehension. A program slicer identifies the parts of a program that are relevant to a given variable reference (i.e., all statements that can influence the resulting value). The resulting slice can then help R programmers and researchers to understand the program by reducing the amount of code to be considered. Furthermore, the slice can help speed up subsequent analyses, by reducing the amount of code to be analyzed.

Objective:
We present flowR, a novel program slicer and dataflow analyzer for the R programming language.
Given R code and a variable of interest, flowR can return the resulting slice as a subset of the program or highlight the relevant parts directly in the input.
Currently, flowR provides a read-eval-print loop, a server connection, and a rudimentary integration into the R language extension for Visual Studio Code which allows to interactively generate and investigate slices. flowR is available as a docker image.

We focus on the R programming language because even though it has a huge, active community, especially in the area of statistical analysis (ranking 7th on the PYPL and 17th on the TIOBE index), the set of existing tools to support R users is relatively small. To our knowledge, there is no preexisting program slicer for R.
Besides the RStudio IDE and the R language server, the {lintr} package and the {CodeDepends} package perform static analysis on R code.
However, all these tools rely on simple heuristics like XPath expressions on the abstract syntax tree (AST), causing their results to be imprecise, limited, and sometimes wrong.
Therefore, we consider flowR's dataflow analysis to be a valuable contribution to these tools by improving their accuracy.

Method:
flowR uses a five-step pipeline architecture, starting with the parser of the R interpreter to build an AST of the program. After normalizing the AST, the dataflow extraction works as a stateful fold over the AST, incrementally constructing the graph of each subtree. The calculation of the program slice reduces to a reachability traversal of the dataflow graph which contains the uses and definitions of all variables. Finally, the slice is either reconstructed as R code or highlighted in the input.

Limitations:
Currently, flowR neither handles R's reflective capabilities (e.g., modifying arguments, bodies, and environments at run-time), nor its run-time evaluation with eval. Moreover, flowR does not perform pointer analysis, which causes vectors and other objects to be treated as atomic (i.e., flowR does not differentiate access to individual elements).

Results:
Using real-world R code (written by social scientists and R package authors) shows that flowR can calculate the dataflow graph (and the respective slice for a given variable reference) in around 500ms. Moreover, averaging over every possible slice in the dataset, we achieve an average line reduction of 86.17%[±9%] (i.e., when slicing for a variable, the relevant program parts are only around 14% of the original lines).

---

1 Weiser. Program Slices. Michigan, 1979.

Slot length:

Reproducible Packaging / 13

Collaborative software development to avoid counterproductive incentives

Authors: Fynn Freyer[1]; Marie Bittiehn[1]; Paul Wolk[2]

Co-authors: Berit Haldemann [2]; Marie Lataretu [2]; Kathrin Trappe [2]; Stephan Fuchs [2]; Piotr Wojciech Dabrowski [1]

[1] *HTW Hochschule für Technik und Wirtschaft Berlin*

[2] *Robert Koch-Institut*

Corresponding Authors: fynn.freyer@htw-berlin.de, piotr.dabrowski@htw-berlin.de

Software development and research are naturally driven by different incentives. While the main aim of software development should be the generation of a robust and easily maintainable product, the daily life of a researcher is dominated by the quest for insightful analysis results and publication deadlines.

We present an approach to solving those conflicting incentives through collaborative development that we have used to develop several research software products. In this approach, two different entities work together closely: The bioinformatics support unit of the Robert Koch Institute (RKI) and the applied computer science bachelor's and master's study programs of the HTW Berlin University of Applied Sciences.

The bioinformatics support unit provides consulting and analyses to other research units within the RKI. The bioinformaticians working there are fully qualified and experienced computer scientists with the knowledge and skills necessary for high-quality software development. Their performance however is primarily measured by the quality and speed of the analyses they perform for the internal customers. Accordingly, prior experience with in-house software development has shown that clean software development practices are hard to sustainably implement in this environment.

On the other hand, the HTW has access to a large number of computer science students with a high interest in software development in and of itself. Those students in their diverse projects and final theses are fully shielded from pressures resulting from the requirements of biological research projects at the RKI. As such, they are a perfect resource for developing research software that is solely focused on robustness and maintainability. However, there are two challenges to overcome when working with computer science students who want to develop research software: Firstly, access to realistic questions and data are hard to come by, and developing software that solves a self-posed problem using simulated data is not very satisfying. And secondly, students are only available during the short time-frames defined by the duration of their projects.

Given that those challenges and opportunities show a high potential for synergy, RKI and HTW have established a cooperation for research software engineering. In short, the daily challenges and requirements concerning analysis software are discussed in semi-regular joint meetings. Work packages and research questions (such as the comparison of different algorithms) are jointly defined and prioritized. Then, those are given out to students at the HTW in the form of projects or final theses. The results are continuously combined and evaluated at the RKI, and flow back into the work package definitions. In order to create a sustainable and continuous environment in which those work packages can be integrated, a single position is co-financed by the RKI and the HTW to supervise and coordinate the entire process.

In this way, a new quality control pipeline was developed that is currently productively used at the RKI, a legacy analysis pipeline was refactored and containerized, and a novel analysis pipeline is being developed. We hope that our experiences –both positive and negative –will encourage others to enter and profit from similar synergetic relationships.

Slot length:

Poster Session / 14

Helmholtz Blablador: An Inference Server for Scientific Large Language Models

Author: Alexandre Strube[1]

[1] *Helmholtz AI*

Corresponding Author: a.strube@fz-juelich.de

Recent advances in large language models (LLMs) like chatGPT have demonstrated their potential for generating human-like text and reasoning about topics with natural language. However, applying these advanced LLMs requires significant compute resources and expertise that are out of reach for most academic researchers. To make scientific LLMs more accessible, we have developed Helmholtz Blablador, an open-source inference server optimized for serving predictions from customized scientific LLMs.

Blablador provides the serving infrastructure to make models accessible via a simple API without managing servers, firewalls, authentication or infrastructure. Researchers can add their pretrained LLMs to the central hub. Other scientists can then query the collective model catalog via web or using the popular OpenAI api to add LLM functionality in other tools, like programming IDEs.

This enables a collaborative ecosystem for scientific LLMs:

- Researchers train models using datasets and GPUs from their own lab. No need to set up production servers. They can even provide their models with inference happening on cpus, with the use of tools like llama.cpp.
- Models are contributed to the Blablador hub through a web UI or API call. Blablador handles loading models and publishing models for general use.
- Added models become available for querying by other researchers.

A model catalog displays available LLMs from different labs and research areas.

Besides that, one can train, quantize, fine-tune and evaluate LLMs directly with Blablador.

The inference server is available at http://helmholtz-blablador.fz-juelich.de

Slot length:

Hackathon: Making Replication Packages Usable (Again) / 15

Hackathon: Making Replication Packages Usable (Again)

Authors: Jacob Krüger[1]; Sebastian Nielebock[2]

[1] *Eindhoven University of Technology, The Netherlands*

[2] *Otto-von-Guericke-University Magdeburg, Germany*

Corresponding Authors: j.kruger@tue.nl, sebastian.nielebock@ovgu.de

Much software engineering (SE) research results in research software artifacts (RSAs), which can be contributions themselves or means to obtain other contributions. However, as research evolves and researchers move on, RSAs tend to be abandoned and become unusable due to a lack of maintenance. As a consequence, other researchers require a lot of effort to re-implement the RSA based on the descriptions in the corresponding publication. Attempts to enhance transparency and artifact availability distilled best practices for sharing artifacts, such as artifact evaluations, badges, and persistent repositories. Still, these attempts are not perfect and RSAs suffer from problems like a lack of documentation, unclear system requirements, or outdated dependencies.

We want to conduct a hackathon as an enjoyable experience for interested participants to see whether, which, and how many artifacts they can reuse from previous publications. Specifically, we will collect a sample of publications (approx. 30) from recent major SE venues, such as ASE, ICSE, and FSE, in which the authors shared RSAs. We will select these so that technical requirements and re-use efforts allow an in-principle reuse within three hours on personal computers. Moreover, we will select RSAs with different badges (e.g., validated vs. non-validated) and provide RSAs targeting different SE domains (e.g., testing, static analysis, refactoring, repair) to motivate many participants. The hackathon is accompanied by an anonymous online survey comprising a minimal set of demographic and background questions as well as the RSA's ReadMe together with a reflection section on its reuse. Single participants or groups (depending on the number of participants) are asked to reproduce the steps in the ReadMe file and replicate the corresponding results. Within the survey, they should document their process and problems by checking whether they can reproduce the single steps in the ReadMe, shortly describing additional steps to achieve reproduction, or issues hindering them. Moreover, participants can update an RSA themselves to make it work (optional), which we track by using version control systems (e.g., `git`). This way, we are collecting steps and best practices for making artifacts (re-)usable as well as identifying RSAs' properties that facilitate or challenge their reuse. We will summarize and synthesize the findings from the surveys and discuss the results with the participants in a separate session. If enough participants are interested in the hackathon, we may be able to publish the findings to guide future RSA sharing. Independently of this case, we will share the synthesized results with the participants; at least during the discussion session and via a report (ideally a published paper) distributed afterwards. If we obtain improved documentations or implementations, we plan to share these with the original RSA authors to consider for updating their repositories. Overall, we hope that this design makes the hackathon an enjoyable experience, with the potential to contribute to new insights as well as improvements to existing RSAs. Since this hackathon is open for practitioners, we hope to get insights on closing the gap between research artifacts and their practical usage.

Slot length:

other(help with comment)

RSE in Digital Humanities / 16

Research Software Engineering in NFDI4Objects: Community building, implementation of FAIRification Tools and scripting in Digital Archaeology

Authors: Fabian Fricke[1]; Florian Thiery[2]; Lutz Krister Schubert[3]

Co-authors: Agnes Schneider [4]; Jürgen Landauer [4]

[1] *Deutsches Archäologisches Institut*

[2] *Leibniz-Zentrum für Archäologie (LEIZA)*

[3] *University of Cologne*

[4] *CAA Deutschland*

Corresponding Authors: lschube4@uni-koeln.de, florian.thiery@leiza.de

Research Software plays an increasing role in the context of Humanities and, specifically, Archaeology to support the analysis of the vast and ever-growing data. As more and more disciplines come together and perform advanced analyses (e.g., with ancient DNA analysis), the demand for reproducible and testable results becomes more serious. So far, most tools have been created ad-hoc to test a hypothesis, but this does not comply with modern objective research practices. Instead, well-designed, and proven tools and methods are needed that allow reproducible and well-structured results. Tools must thereby be equally accessible and FAIR as the data itself, in compliance with the standard right of access to and participation in culture.

NFDI4Objects (N4O) is a broad community dealing with material remains of human history, the FAIR and CARE principles as well as FAIR4RS. The goal is to integrate the community into so-called

Community Clusters to strengthen *Software as Research Data*, *Publication and Citation of Research Software* as well as the *RSE profile*.

This paper presents the community participation possibilities, N4O FAIRification Tools (e.g., Alligator, AMT, SPARQL Unicorn) and examples from *Computational Archaeology* (e.g., R and Python scripts, AI) in the context of CAA-DE.

During the last German chapter CAA conference in Würzburg (September 2023), multiple software tools were presented and discussed, such as for modelling stratigraphy (implemented in Python), cluster analysis for archaeological finds (implemented in R), using AI techniques for detection of archaeological sites on satellite images or classification of Celtic coins. All these approaches were designated using different tools, programming methods and methodologies. It could be noted that very few of them adhered to (Research) Software Engineering principles, making it difficult for any uptaker to understand or re-use the code. What is worse, few were published or made accessible, as the results (aka the data) were deemed more important than the means for generating them. This impacts reproducibility and therefore, reduces the value and credibility of the results. Most tools were developed for analysis and, therefore, have a notion of "quick hacks", which developed into more complete programs as the questions asked started to develop with the analysis. This also led to a lack of re-use of existing tools and methods.

These challenges are not new in the context of IT and are representative of any scientific code development, but awareness of their relevance for good scientific work is slowly rising. Areas new to IT are however more susceptible to these pitfalls. It is therefore even more relevant to identify these issues from the beginning and develop and teach good RSE principles with these communities; to demonstrate the relevance of said principles and to not see them as a burden but as a potential for continuation and improvement of research.

In this paper we want to highlight the challenges and approaches to engage the archaeological community in Research Software Engineering.

Slot length:

Sustainable Organisation / 17

Emerging Heterogeneous Computing Architectures Every Software Engineer Should Know (Compilers and DSLs)

Author: Babar Khan[1]

[1] *TU Darmstadt, Germany*

Corresponding Author: babar.khan@tu-darmstadt.de

Software engineers communicate with hardware through a language known as an instruction set architecture (ISA). However, the conclusion of Dennard scaling and Moore's Law implies that this traditional ISA is being replaced by complex heterogeneous ISAs. In light of this, the expertise required to develop new compilers and create domain-specific languages will be crucial for future software engineers. Therefore, this talk will focus on addressing the challenges posed by emerging computing architectures and research related to abstracting the complex ISA. As a use-case, the talk will present an open-source tool that recently received the Best Paper Award at one of the leading computer architecture conferences.

Slot length:

other(help with comment)

Local RSE-related Organisations / **18**

NHR - HPC for Scientists

Author: Doerte Sternel[1]

Co-author: Barbara Diederich [1]

[1] *NHR-Verein, e.V.*

Corresponding Author: doerte.sternel@nhr-verein.de

NHR is an alliance of university high-performance computing centers funded by the federal and state governments. NHR offers not only computing resources, but also consulting and training for scientists at German universities.
In our presentation, we will give a brief overview of NHR and the services it offers to scientists.

Slot length:

RSE Research / **19**

Exploring the RSE Landscape at HZDR: Initial Approach and First Results

Authors: Fredo Erxleben[1]; Thomas Foerster[2]

[1] *Helmholtz-Zentrum Dresden-Rossendorf*

[2] *HZDR*

Corresponding Author: f.erxleben@hzdr.de

The presentation outlines the initial approach and the preliminary results of a comprehensive survey aimed at mapping the Research Software Engineer (RSE) landscape within the Helmholtz-Zentrum Dresden-Rossendorf (HZDR). The primary objective of this survey is to provide a comprehensive overview of the current status of RSE activities at HZDR, including the diverse range of RSE projects being undertaken. By examining the emerging trends and challenges within the institution, this research aims to equip decision-makers with essential insights to facilitate the future establishment and training of RSEs at HZDR.

Slot length:

Sustainable Organisation / **20**

Sustainable open-source research software by founding a start-up: A LinkAhead story

Author: Florian Spreckelsen[1]

Co-authors: Alexander Schlemmer [1]; Daniel Hornung [1]; Henrik tom Wörden [1]; Sina Rohde [1]; Thomas Weiß [1]; Timm Fitschen [1]

[1] *IndiScale GmbH*

Corresponding Author: f.spreckelsen@indiscale.com

Often, software developed within a research group to solve specific problems evolves into an integral tool, sometimes not only for the original group, but also for other groups and institutions where it may be adapted. A well-known problem, however, is to guarantee long-term stability and maintenance, as well as the further development of the software. A lack of (permanent) funding as well as high personnel fluctuation, among others, pose challenges

In this talk, we will present how maintenance and development of such a research software, namely our open-source data-management toolkit LinkAhead (formerly known as CaosDB), could be solved by founding IndiScale, a company that provides services around this software. We will explain which considerations lead to the decision to found and why LinkAhead was well-suited for founding a supporting Start-up company.

We will also give an overview the problems to be faced, as well as the different requirements to project management and day-to-day work compared to the life as researchers and research software engineers in academia.

Slot length:

21

RSE-related Meet-ups - HackyHour, DataDojo & co.

Author: Markus Ankenbrand[1]

[1] *University of Würzburg, Center for Computational and Theoretical Biology, BioMedical Data Science*

Corresponding Author: markus.ankenbrand@uni-wuerzburg.de

I co-organize the monthly HackyHour at the University of Würzburg. A social gathering where people interested in computational tools for research meet and exchange ideas. It is also aimed to provide help for students and researchers with specific problems. I also organizers the regular Data Dojo at my organization. In the Data Dojo a pre-selected data set is collectively analyzed, with everyone taking turns at the keyboard.

I would love to meet organizers of similar events or people interested in starting such meet-ups. We could discuss the pros and cons of different formats, how to advertise and sustainably organize these events. Further, we can identify common pitfalls and discuss ways to connect the existing initiatives all across Germany (and beyond).

Slot length:

Workshop (1h)

Poster Session / 22

Refactoring and isolation data pipelines through the use of software containerization and continuous integration

Author: Benjamin Bruns[1]

[1] *Forschungszentrum Jülich GmbH*

Corresponding Author: b.bruns@fz-juelich.de

At the IAS-8 institute of Forschungszentrum Jülich, the accurate and complete collection of measurement and environmental data is essential for subsequent analyses and modeling in many projects. Although the Bayeos server (https://github.com/BayCEER/bayeos-server) used at FZJ provides an open and standardized data platform for such data, the import and transformation of data from different sources is often difficult in terms of provision, traceability and subsequent adjustments. To address this problem, a flexible import and transformation pipeline for time series data was developed based on Python and a PostgreSQL-based integration database. There is a clean separation of import, transformation and aggregation processes, which also allows for easy customization. Each individual step of the defined pipeline runs as a container in a Docker environment. There is a template for a basic pipeline, which can be easily customized to define additional pre- and post-processing steps. This template has been successfully adapted for different existing data pipelines. Once this has been done, the containers are built automatically using the CI/CD pipeline of the DevOps platform Gitlab. In addition, Gitlab's own container registry ensures easy deployment and updating of the pipeline elements.

Slot length:

RSE Communities / **23**

US-RSE: Today's successes and tomorrow's challenges

Author: Daniel Katz[1]

Co-authors: Ian Cosden [2]; Jeffrey C. Carver [3]

[1] *University of Illinois Urbana-Champaign*

[2] *Princeton University*

[3] *University of Alabama*

Corresponding Author: d.katz@ieee.org

The US Research Software Engineer Association (US-RSE) was formed in 2018 as a grassroots effort by a handful of motivated volunteers who focused on community building. Since its inception, US-RSE has hosted dozens of community calls, virtual and in person workshops, and other events that bring together RSEs. With over 2000 members today, it has been successful in much of its mission: 1) community, 2) advocacy, 3) resources, and 4) diversity, equity, and inclusion. Many things have changed over this time, such as having three versions of the website, two financial sponsors, and now professional staff and a conference. These changes, and the five years of existence and activities, however, lead to challenges as well, such as successfully mixing paid and volunteer work, avoiding or dealing with volunteer burnout and the concomitant need for new volunteers, moving from building community to providing services to that community, and moving from describing problems to collectively solving them. This talk will discuss both the successes and challenges, and is intended to lead to comparative discussion with members and leaders of other RSE organizations, where we hopefully can learn from each other.

Slot length:

Poster Session / **24**

Software Management Plans as a Path for Sustainable and Reproducible Research Software –Potentials, Discussions and Obstacles in Dealing with Code in Science

Authors: Michael Franke[1]; Yves Vincent Grossmann[1]

[1] *Max Planck Digital Library*

Corresponding Author: franke@mpdl.mpg.de

Research software is receiving increasing attention. Sustainability, reproducibility, publishing and recognition are four of the key issues that are currently being discussed. Software management plans can be an asset in all aspects. In our poster we will discuss the four topics with reference to SMPs. For this, we will analyse the status quo of SMP templates and discussion points. We will also show possible ways towards producing code usable and accepted by the scientific community. The aim is to develop a further iteration of the RDMO SMP catalogue that can be be discussed together in the field.

Slot length:

RSE Practices / **25**

Agile Software Engineering of Research Data Management Infrastructures

Authors: Mahmoud Hassan[1]; David Glück[1]; Klaus Tochtermann[1]

[1] *ZBW - Leibniz-Informationszentrum Wirtschaft*

Corresponding Authors: mhassan@zbw-workspace.eu, dglueck@zbw-workspace.eu

The BERD@NFDI consortium aims to establish a research data management platform for economics within the German National Research Data Infrastructure (NFDI). This platform will host diverse resources such as research data from areas like marketing, machine learning models, and company reports, involving various partner institutions and user communities. This results in an agile, user-driven requirements engineering process.

In this talk, we will focus on our software engineering approach in the context of this process and will share our experiences in the following areas: First, the infrastructure is managed in a cloud solution for which we have built a multi-tier technology stack to best support our continuous development and deployment approach; second, for the research data management we are using the repository software InvenioRDM that is built on the open source Invenio framework.

After a short introduction of our cloud-based technology stack, our agile software development process and InvenioRDM, the talk will explore in more depth our adaptation and extension of this framework reflecting the specific needs of the BERD user community. Of particular importance is InvenioRDM's flexibility to support the development of domain-specific user interfaces and custom metadata models aligned with the FAIR principles. Its modular and domain-driven architecture, characterized by distinct layers encompassing data access, service, and presentation, renders the code structure easily comprehensible. This layered architecture also facilitates the construction of custom modules, allowing for seamless extension according to the specific functionalities desired by our user community, for instance for adding new types of research data, implementing fine-grained search capabilities, and enabling quality checks for the data presented in the platform. We discuss the advantages and drawbacks of this flexibility, including code complexity and technical debt, and how we address these issues through quality assurance measures like comprehensive testing and GitLab-based deployment.

Slot length:

Poster Session / **26**

Streamlining Metadata Handling in Research Software Engineering

Authors: Anton Pirogov[1]; Mustafa Soylu[1]

Co-authors: Stefan Sandfeld [1]; Volker Hofmann

[1] *Forschungszentrum Jülich*

Corresponding Authors: a.pirogov@fz-juelich.de, m.soylu@fz-juelich.de

Modern research is heavily dependent on software. The landscape of research software engineering is evolving at a high pace, and the effective handling of metadata plays a pivotal role in ensuring software discoverability, reproducibility, and general project quality. Properly curating metadata can, however, become a time-consuming task, while manual curation is error prone at the same time. This poster introduces two new tools for streamlining metadata management: fair-python-cookiecutter and somesy.

The fair-python-cookiecutter is a GitHub repository template which provides a structured foundation for Python projects. The template provides researchers and RSEs with support in meeting the increasing demands for software metadata during development of Python tools and libraries. By cloning and applying the template to their projects, developers can benefit from the incorporated best practices, recommendations for software development, and software project metadata to ensure quality and facilitate citation of their work. The fair-python-cookiecutter is aligned with and inspired by standards like DLR Software Engineering Guidelines, OpenSSF Best Practices, REUSE, CITATION.cff, CodeMeta. Furthermore, it uses somesy to enhance software metadata FAIRness. The template comes with detailed documentation and thus offers an accessible framework for achieving software quality and discoverability within academia.

Somesy (software metadata synchronization) provides a user-friendly command-line interface that assists in synchronization of software project metadata. Somesy supports best-practice metadata standards such as CITATION.cff and CodeMeta and automatically maintains metadata, such as essential project information (names, versions, authors, licenses), consistently across multiple files. This ensures metadata integrity and frees additional time for developers and maintainers to focus on their work.

https://github.com/Materials-Data-Science-and-Informatics/fair-python-cookiecutter
https://pypi.org/project/somesy/

Acknowledgements: The presented content was created by the FAIR data commons & Hub Information of the Helmholtz Metadata Collaboration (HMC) at Forschungszentrum Jülich. HMC is an Incubator platform of the Helmholtz Association within the framework of the Information and Data Science strategic initiative

Slot length:

Poster Session / 27

Best practice made easy: Deploying tools for FAIR research software development

Author: Anton Pirogov[1]

Co-authors: Stefan Sandfeld [1]; Volker Hofmann

[1] *Forschungszentrum Jülich*

Corresponding Author: a.pirogov@fz-juelich.de

Sustainable software development and metadata practices are crucial for making research software FAIR. Undeniably, this requires an initial investment of time and effort for researching and adopting best practices, as well as for regular maintenance tasks. This impairs the bottom-up adoption of best practices by RSEs. In this talk we present two complementary tools addressing this challenge:

The command-line tool somesy1 organizes and synchronizes software project metadata across multiple required or recommended files that are typically used in a software project. It is designed for easy integration into the development workflow, reducing the overhead for software metadata management.
The fair-python-cookiecutter2 is a git repository template providing a well-structured foundation for new Python projects. The detailed documentation also turns it into a hands-on educational resource. Besides combining various state-of-the-art tools, it includes somesy to simplify software metadata management. The template follows software engineering practices recommended by DLR and OpenSSF. Furthermore, it supports relevant metadata standards such as REUSE, CITATION.cff and CodeMeta.

We will present both tools in this demo.

Slot length:

Lessons learned and applied / **28**

Best practice made easy: Deploying tools for FAIR research software development

Authors: Anton Pirogov[1]; Mustafa Soylu[1]

Co-authors: Stefan Sandfeld [1]; Volker Hofmann

[1] *Forschungszentrum Jülich*

Corresponding Author: m.soylu@fz-juelich.de

Sustainable software development and metadata practices are crucial for making research software FAIR. Undeniably, this requires an initial investment of time and effort for researching and adopting best practices, as well as for regular maintenance tasks. This impairs the bottom-up adoption of best practices by RSEs. In this talk we present two complementary tools addressing this challenge:
The command-line tool somesy1 organizes and synchronizes software project metadata across multiple required or recommended files that are typically used in a software project. It is designed for easy integration into the development workflow, reducing the overhead for software metadata management.
The fair-python-cookiecutter2 is a git repository template providing a well-structured foundation for new Python projects. The detailed documentation also turns it into a hands-on educational resource. Besides combining various state-of-the-art tools, it includes somesy to simplify software metadata management. The template follows software engineering practices recommended by DLR and OpenSSF. Furthermore, it supports relevant metadata standards such as REUSE, CITATION.cff and CodeMeta.

Slot length:

Sustainable Organisation / **29**

Nachhaltige Entwicklung und Wartung von Codeerweiterungen und Simulations-Setups für quelloffene Strömungssimulationssoftware

Author: Ronald Lehnigk[None]

Co-author: Fabian Schlegel [1]

[1] *Department of Computational Fluid Dynamics, Helmholtz-Zentrum Dresden-Rossendorf, Germany*

Corresponding Author: r.lehnigk@hzdr.de

Eine für die numerische Simulation von Strömungen sehr beliebte und erfolgreiche Software ist die quelloffene Software der OpenFOAM Foundation, welche sowohl in der Industrie als auch im akademischen Umfeld Anwendung findet. Forschungsgruppen, die keine reine Inhouse-Entwicklung leisten können oder anstreben, gewährt sie eine optimale Basis um eigene Ideen und Konzepte in einer transparenten Umgebung effizient testen zu können. Obwohl der Wartungsaufwand im Vergleich zu einer Eigenentwicklung insgesamt erheblich geringer ist, müssen Erweiterungen dennoch gepflegt werden um diese mit dem jeweils aktuellen Stand des Hauptrelease kompatibel zu halten. Die damit verbundene Arbeit erlangt umso größere Bedeutung, wenn Erweiterungen im Sinne der FAIR-Prinzipien zusammen mit wissenschaftlichen Publikationen bereitgestellt werden. Als agil entwickelte und intensiv gewartete Software stellt die Software der OpenFOAM Foundation in dieser Hinsicht besondere Anforderungen an die nachgelagerten Entwickler.

Das Helmholtz-Zentrum Dresden –Rossendorf e.V. (HZDR) verfolgt hierbei einen möglichst nachhaltigen Ansatz. Abgeschlossene und zitierfähige Entwicklungen werden entweder in einer eigenen Softwarepublikation veröffentlicht, oder, in enger Abstimmung mit den Kernentwicklern der OpenFOAM Foundation, in das Hauptrelease integriert. Die für die Arbeit an der Erweiterung (Multiphase Code Repository by HZDR for OpenFOAM Foundation Software) geschaffene IT-Infrastruktur zeichnet sich durch einen hohen Automatisierungsgrad aus und bietet Anwendern innerhalb und außerhalb des HZDR eine nützliche Plattform für die Erforschung von numerischen Methoden und Modellen.

Rückgrat der Arbeiten ist die über die Helmholtz Cloud bereitgestellte GitLab-Instanz (Helmholtz Codebase). Darin werden zwei Repositorien gepflegt: Eines für die Codeerweiterung und eines für Setups zur Simulation konkreter Anwendungen (Multiphase Cases Repository by HZDR for OpenFOAM Foundation Software). Zur Sicherung der Qualität und Funktionalität wird die Arbeit in der GitLab-Umgebung von Continuous-Integration-Pipelines (CI) begleitet, in deren Rahmen unter anderem statische Code-Checks, Build-Tests und Testläufe automatisiert vorgenommen werden. Für die Verwendung in CI-Pipelines sowie die lokale Entwicklung der Erweiterung wird die Installation als Container (Docker) bereitgestellt. Reine Anwender können auf die Installation per Debian-Paket zurückgreifen. Die zitierfähige Veröffentlichung des Quellcodes erfolgt mit jeder wissenschaftlichen Publikation im Rossendorf Data Repository (RODARE). Die Verwendung des Workflowmanagementsystems Snakemake ermöglicht skalierbare Validierungsläufe. Um die Portierbarkeit der Entwicklungen zu verbessern konzentrieren sich jüngere Arbeiten auf die Bereitstellung der Software als HPC-Container (Apptainer) für die Anwendung auf Hochleistungsrechnern. Dieser Beitrag gibt einen Überblick über die genannten Elemente der Umgebung und deren Zusammenspiel.

Slot length:

Research Software for Computing and Visualising Text / 30

OCR(-D)4all - An easy to use and highly adaptable open-source solution for automatic text recognition of historical printings and manuscripts

Authors: Christian Reul[1]; Herbert Baier Saip[1]; Maximilian Nöth[1]

[1] *Zentrum für Philologie und Digitalität*

Corresponding Authors: maximilian.noeth@uni-wuerzburg.de, herbert.baier@uni-wuerzburg.de

Despite the various challenges in automatic text recognition for printed (OCR) and handwritten (HTR) material, great progress has been made during the last decade. Several milestones have been reached regarding the actual text recognition step but also in layout analysis and pre- and postprocessing. Additionally, free open-source implementations of related tools and algorithms are released constantly. While these allow tackling highly heterogeneous use-cases ranging from mass full-text digitisation in libraries to the processing of individual documents, including specific production of

training data (Ground Truth, GT) and subsequent training of deep learning OCR/HTR models, they rarely offer standardized interfaces and a low barrier of entry for non-technical users.

To solve the issue of easy-to-use, flexible, connectable, and sustainable combination and application of current and future individual technical OCR/HTR solutions we introduce the open-source tool OCR4all, which in turn leverages the open-source OCR/HTR framework OCR-D. In the following we discuss how users can benefit from the powerful combination of the two.

Whereas OCR-D concerns on the standardized implementation of single-step processors, OCR4all aims at enabling any user –even those without technical background –to perform OCR/HTR completely on their own and in great quality, while also offering tools to manually generate training data in order to train more performant work-specific models and consequently improve the output of the fully automated OCR/HTR processors.

To combine both approaches we engineered interfaces between OCR-D tools and OCR4all based on the OCR-D specifications. For a very flexible integration of different OCR and especially OCR-D processors, OCR4all relies on the Java Service Provider Interface. OCR-D processors, which are written in Python, are plugged into OCR4all as containerized service providers that implement the required interfaces, either through a simple manual configuration or fully automatically. The latter is achieved by leveraging the OCR-D-Tool-JSON which contains all necessary information about input/output relationships and available parameters and is mandatory for all OCR-D processors.

Due to the above described adaptations and the thereby extended flexibility, OCR4all can now be applied to an even more heterogeneous selection of materials and use cases. Its main focus still lies on the interactive high quality processing of challenging early printings and manuscripts by non-technical users, including correction and GT production and consequently material-specific training. However the applicability of OCR4all for mass digitization, e.g. in libraries and archives, has vastly improved.

Slot length:

Poster Session / 31

Computational workflow to build a data-driven model of a „Virtual Parasite“

Author: Lucas Fortune[None]

Co-author: Philip Kollmannsberger [1]

[1] *Heinrich-Heine-Universität Düsseldorf*

Corresponding Author: lucas.fortune@hhu.de

In this project, we will build “virtual parasites”from high-resolution image data as basis for a precise data-driven mechanical understanding of parasite biophysics in the context of the DFG priority programme “Physics of Parasitism”.

The parasitic life cycle involves a multitude of physical interactions with the host microenvironment during stages of motility and adhesion. The shape and elasticity of unicellular parasites are largely defined by their cytoskeleton. In parasitic kinetoplastids such as Trypanosoma brucei, the cytoskeleton includes a subpellicular array of microtubule filaments that forms a corset around the entire cell. How exactly the interaction between the beat of the flagellum, which is attached to the cytoskeleton and winds around the cell, and the mechanical response of the cell body, gives rise to the intricate rotational motility patterns of T. brucei is not known.

To answer this question, a detailed structural and mechanical model of the microtubule cytoskeleton and the interior of the cell, which together define its elasticity, is needed. We will first perform a complete semantic segmentation (pixel-level annotation) of electron tomography image volumes of different developmental stages of T. brucei using a deep learning workflow. We will then apply

automated tracing of microtubule filaments and semi-supervised instance segmentation of the cell interior to arrive at a complete 3D structural model of the cell. This model will be converted to an annotated volume mesh amenable for finite element analysis
and subjected to in-silico deformation test to validate the model against experimental data. By adapting the workflow to other image data, similar mechanobiological phenomena in other model parasites can be studied. Our data-driven approach will enable new ways of understanding the physics of parasitism by connecting imaging with computer simulations.

The complete workflow as well as the image data and model will be made available to collaborators within the DFG Priority Programme "Physics of Parasitism". A web-based collaboration platform for visualizing and exploring the image and model data will be developed and hosted alongside the data. All code and accompanying interactive Jupyter notebooks for image analysis and neural network training will be under version control and accessible from a gitlab instance. Pre-trained networks will be available for download, as well container definition files to run the workflows in different computational environments. Image analysis code that could be relevant for other researchers within the Physics of Parasitism priority programme will be documented and made available on the same platform. We will ensure publication of code and data under the FAIR principles in all collaborating projects by working closely with the research data management units and libraries.

Slot length:

Analysis and Visualization / **32**

PostWRF: Interactive tools for the visualization of the WRF and ERA5 model outputs

Author: Amirhossein Nikfal[1]

[1] *Forschungszentrum Jülich*

Corresponding Author: a.nikfal@fz-juelich.de

PostWRF is an open-source software toolkit to facilitate the main visualization tasks and data handling for the Weather Research and Forecasting (WRF) model outputs. The toolkit is mostly written in NCL and Shell, with a namelist that resembles the WRF or WPS namelists. Besides the visualizations, PostWRF provides WRF-NetCDF to GeoTIFF conversion for GIS applications, ERA5-NetCDF reanalysis data plotting and extraction. The primary purpose of PostWRF is to benefit the environmental researchers (both experienced and inexperienced) to make use of the WRF model simulations, in a straightforward and efficient way, without dealing with coding and syntax errors. Since the WRF model simulates most aspects of a full atmospheric model in the regional scale, the toolkit can also be used as an educational aid in meteorological and environmental science. PostWRF is available on GitHub (https://github.com/anikfal/PostWRF), provided with HTML documentations (https://postwrf.readthedocs.io/en/master) and guided examples.

Slot length:

Workshop (1h)

Workflowmanagement for Parallel Computing / **33**

Uncharted Waters Ahead. Moving Legacy Software Infrastructure to Kubernetes

Authors: Adrian Sturm[1]; Michelle Weidling[2]; Stefan Hynek[1]

[1] *Niedersächsische Staats- und Universitätsbibliothek Göttingen. Forschung und Entwicklung*

[2] *Niedersächsische Staats- und Universitätsbibliothek Göttingen. Digitale Bibliothek*

Corresponding Authors: sturm@sub.uni-goettingen.de, stefan.hynek@uni-goettingen.de

Based on the precondition of the availability of Kubernetes as a Service, this talk is going to show why the migration of legacy software projects to a Kubernetes cluster is a worthwhile undertaking and how it can succeed, including the decisions made for the tools and the lessons learned on the way.
Starting from the current state with multiple deployment tools and environments (VMs, dedicated servers, Puppet, manual installs), we briefly cover the main problems arising from this historically grown structure.
We will then introduce a toolchain of CNCF Graduated software comprising Helm Charts as a standardized format for the deployment configuration of an application and ArgoCD as a deployment tool for continuously delivering applications to a Kubernetes cluster in a declarative and GitOps manner.
We will then see how this new toolchain for deployment and application hosting resolves or mitigates the aforementioned problems.
The second part of the talk will give a brief introduction of supporting services running inside the cluster that contribute to having a controlled, stable and easy to maintain environment for application deployments. This will cover possible solutions for certificate issuing, secrets management, logging and error tracking.
We will close the talk with an outlook on how a software development team could adapt to the new workflow and open a discussion on possible problems (e.g. limited resources) and coping strategies.

Slot length:

Metadata for Research Software / 34

Requirements for Metadata of Energy Research Software

Author: Stephan Alexander Ferenz[1]

Co-authors: Oliver Werth [2]; Astrid Nieße [1]

[1] *Carl von Ossietzky Universität Oldenburg*

[2] *OFFIS e.V. –Institute for Information Technology*

Corresponding Author: stephan.ferenz@uni-oldenburg.de

Energy research software (ERS) is used in energy research for multiple purposes like visualization of processes and values, e.g., power quality, (co-)simulation of smart grids, or analysis of transition paths for energy systems. Within an exemplified research cycle, this software is often fundamental for producing new research results while it can also present a result of performed research. (1)
Metadata have shown to be one of the success factors for the so-called FAIRification of research software, especially to improve findability and, thus, reusability of research software (2), (3). To reach a high interoperability of metadata as part of the FAIRification these metadata should follow a defined schema with extensive reuse of relevant metadata elements from other schemas. Within the energy domain some approaches to collect metadata for energy research software already exist (e.g., the openmod wiki, or the Open Energy Platform). However, none of these approaches uses a formalized and interoperable metadata schema to open up the approach for further reuse for FAIR ERS.
As first step to develop a metadata schema requirements have to be gathered on which information should be included in the metadata (4). Therefore, the goal of our work is to gather these specific requirements for a metadata schema for ERS.
To this end, we follow a qualitative research approach to get relevant requirements from multiple stakeholders: we conducted semi-structured interviews with 36 researchers from different subdomains of energy research, e.g., research on power grids or specific components. The researchers

use different types of software (from scripts over libraries to stand-alone software). Our interviews followed a rough interview guideline, based on the FAIR criteria, structured in five main categories: findability of general fitting ERS, selection of the right ERS for certain research, accessibility, interoperability, and reusability.
The interviews show a diverse field of requirements for ERS due to different reasons. First, depending on the subdomain, the ERS are highly diverse. Second, the scientific backgrounds of energy researchers lead to different requirements, e.g., regarding the choice of programming languages. The interviews show especially a need for information on the community and quality of ERS.
In our talk, we will present the results of our requirement analysis and discuss them with the audience.

References
(1) S. Ferenz, "Towards More Findable Energy Research Software by Introducing a Metadata-based Registry,"in Abstracts of the 11th DACH+ Conference on Energy Informatics, Anke Weidlich, Gunther Gust and Mirko Schäfer, Ed., Springer, 2022. doi: 10.1186/s42162-022-00215-6.
(2) D. S. Katz, M. Gruenpeter, and T. Honeyman, "Taking a fresh look at FAIR for research software," Patterns, vol. 2, no. 3, Mar. 2021, doi: 10.1016/j.patter.2021.100222.
(3) A.-L. Lamprecht et al., "Towards FAIR principles for research software,"Data Sci., vol. 3, no. 1, pp. 37–59, Jan. 2020, doi: 10.3233/DS-190026.
(4) M. Curado Malta and A. A. Baptista, "Me4DCAP V0.1: a Method for the Development of Dublin Core Application Profiles,"Min. Digit. Inf. Netw., pp. 33–44, 2013, doi: 10.3233/978-1-61499-270-7-33.

Slot length:

Poster Session / 35

Using TimescaleDB and Apache Superset to support Co-Simulation Data Exploration

Authors: Alexandro Steinert[1]; Tobias Brandt[1]

[1] *OFFIS e.V.*

Corresponding Author: alexandro.steinert@offis.de

Co-Simulation is a technique commonly used in research to analyse complex systems that are hard to simulated by monolithic simulation models. In co-simulation, different models are used simultaneously to represent a system. Especially in the energy sector, where the systems often have many components, co-simulation is a standard solution. Mosaik is an open source co-simulation tool developed for the energy domain. It allows to couple multiple models very flexibly, wherefore simulations with complex dependencies can be set up. When running a simulation through mosaik, a multitude of data is generated through the connected models. The analysis of this data is difficult to do and requires knowledge about programming libraries such as pandas and Matplotlib. The goal of this work is to introduce software tools and proper documentation to simplify the process of analysing and visualising simulation data for researchers.

This abstract introduces a Timescale database adapter for mosaik, which allows for a more seamless collection of simulation data. This adapter, in conjunction with Apache Superset, allows for analysis and interpretation of co-simulation data from mosaik. The visualisation through Superset allows researchers to easily demonstrate their findings visually and share them with others. This improves the data analysis process and increases the explainability of research findings.

The presented software is developed as a specialized SQL adapter for the Timescale database. TimescaleDB is chosen, as most generated simulation data is timeseries data, where TimescaleDB offers sophisticated support for.

While the adapter is written with TimescaleDB in mind, it also supports generic Postgres databases. The user can choose between multiple table layouts for storing the data. The adapter is open source under the MIT license and thus accessible for anyone to use or modify for their use case.

Additionally, Apache Superset is chosen as the visualisation tool. Superset is an open source data visualisation tool with an interactive web interface. The usage of the visualization component of this work depends on the actual simulation setup. Nevertheless, due to similarities between simulations in the energy domain, helpful defaults and explanations are given in our tutorial for users to have a quick and seamless experience when first using it in conjunction with mosaik.

To conclude, a TimescaleDB adapter was developed for the mosaik co-simulation tool and a tutorial to visualize the data in Apache Superset was written and published. The adapter is easy to use and, through Superset, enables researchers to make their data explainable and accessible with little programming knowledge needed. In future work, the adapter can be expanded by saving simulation metadata to tables in the database. This allows the researchers to gain insights about the simulation itself, such as simulation time, and enables a better understanding and analysis of the results. Furthermore, more tutorials for using the adapter with different visualisation tools such as Grafana can be added.
Sourcecode: https://gitlab.com/mosaik/components/data/mosaik-timescaledb
Tutorial: https://mosaik.readthedocs.io/en/latest/tutorials/apache-superset.html

Slot length:

Poster Session / 36

User experience driven design of the Helmholtz KG User Interface

Authors: Fiona D'Mello[1]; Jens Bröder[1]; Stefan Sandfeld[1]; Volker Hofmann[1]

[1] *Institute for Advanced Simulation (IAS-9), Forschungszentrum Jülich, Germany*

Corresponding Author: f.dmello@fz-juelich.de

User experience (UX) is a critical component when developing an intuitive and informative user interface (UI). According to Garrets 5 elements of UX Design 1 development should orient as per the needs and requirements of use cases and users.

The Helmholtz Knowledge Graph (Helmholtz KG) is being developed since late 2022, as a lightweight interoperability layer between Helmholtz Infrastructures. It aims to connect Helmholtz (meta-)data, which is currently stored in a disconnected and silo-ed fashion within the infrastructures of the respective centers. With aggregating information about Helmholtz's digital assets we intend to increase data findability and offer their full value to scientists, strategists and administrators, in and outside of the association.

In order to create a suitable user interface, we started requirement engineering by defining stakeholders, sketch fictive use cases and extract potential requirements from this exercise. Our first UI, based on these requirements, went online (link) in 10/2023 and provides an easy and systematic access to the data currently in the graph.

In our poster we will present the current state of the Helmholtz KG UI, and how its components relate to the 5 elements of UX Design. We will further give an outlook of our recently started second phase of requirement engineering that includes our stakeholders directly. For this, we conducted multiple structured workshops to establish feedback from differing stakeholder group in our communities. This allows us, to corroborate our work from the initial phase as well as, identify new features for further development. The structure of the workshop was based on the tools and methods available under Service Design 2, extending on the previously mentioned UX design principles.

1 The Elements of User Experience: User-centered Design for the Web and Beyond. (2011). United Kingdom: New Riders.

2 Stickdorn, M., Hormess, M. E., Lawrence, A., Schneider, J. (2018). This Is Service Design Doing. United States: O'Reilly Media.

Acknowledgements: The presented content was created by the FAIR data commons & Hub Information of the Helmholtz Metadata Collaboration (HMC) at Forschungszentrum Jülich. HMC is an Incubator platform of the Helmholtz Association within the framework of the Information and Data Science strategic initiative

Slot length:

AI/ML Research Software / 37

Deep Reinforcement Learning in agent-based model AgriPoliS to simulate strategic land market interactions

Authors: Changxing Dong[1]; Ruth Njiru[None]; Franziska Appel[None]; Alfons Balmann[None]

[1] *Leibniz Institute for Agricultural Development in Transition Economies (IAMO)*

Corresponding Author: dong@iamo.de

AgriPoliS (Agricultural Policy Simulator) is an agent-based model for simulating the development of agricultural regions, focusing on the structure changes under economical, ecological and societal factors. The farms in the region are modeled as agents in AgriPoliS, which interact with each other through different markets, most importantly, land market. Every year the agents make their decisions about bidding for new land plots, stable and machine investments, and production processes through mixed linear programming (MIP) optimizations, to maximize their profit, where the focus is only for the current year with no regard for the future implication of these decisions. In this work, we enhance the agents with Deep Reinforcement Learning, giving them the ability to develop strategic instead of myopic decisions to maximize their long-term profits within the simulation period through strategic bidding behavior in the land market.

As the first step, only one agent is enhanced with Reinforcement Learning while the other agents adopt the standard behavior. As we are interested in the effects of the strategic bidding behavior of the agent, we formulate bidding prices as the action space. The state space consists of the state variables of the agents and the region under investigation, which include liquidity, current stables and machines, the distribution of remaining contract duration for rented land, rent prices, spatial distribution of free land plots in the region and the distribution of competitive agents in the neighborhood. Equity capital of the agent is considered as the reward from the environment by taking an action. Using the state variables, the agent chooses the action (bidding price) to present to the land market. Based on the success and/or failure of his bid, the agent proceeds to make investment and production decisions and obtains the results of his decisions. This continues until the end of the simulation period when the cumulative equity capital is accessed.

The learning framework consists of two parts, AgriPoliS and learning algorithm. AgriPoliS, which is implemented in C++, functions as the environment. It takes the action from the learning algorithm, which is implemented in Python, and deliver the states and rewards to the learning algorithm. The communication between the two parts is realized through the message queue system zeorMQ.

Since the action space is continuous, we implemented the DDPG (Deep Determinant Policy Gradient) algorithm with PyTorch. After tuning the learning hyper parameters, the enhanced agent could learn a stable strategy, which varies the relative bidding prices and maximizes the cumulative rewards. The first results are promising, the effect of the agent changing its bidding behavior not only affects its equity capital but also the equity capital of other farms. More algorithms like TD3 (Twin Delayed DDPG) and SAC (Soft Actor-Critic) are under work and we are also interested to resolve the learning stability issue of the algorithms.

Slot length:

Analysis and Visualization / **38**

Structured Experiment Metadata Acquisition Using Adamant: Current State and Concepts

Author: Ihda Chaerony Siffa[1]

Co-authors: Robert Wagner [1]; Markus M. Becker [1]

[1] *Leibniz-Institut für Plasmaforschung und Technologie e.V.*

Corresponding Author: ihda.chaeronysiffa@inp-greifswald.de

Structured and machine-readable experiment metadata enable various analysis and visualizations of the stored data and metadata, which may not be easily achievable with unstructured metadata such as free texts. On top of that, structured metadata increases the findability and re-usability of said metadata (and the data to which the metadata is attached) for other purposes following the spirits of the FAIR data principles. Adamant is a browser-based research data management (RDM) tool, specifically developed to systematically collect experiment metadata that is both machine- and human-readable. It makes use of the JavaScript Object Notation (JSON) schema, where valid schemas can be rendered as an interactive and user-friendly web form. Researchers may create a JSON schema that describes their experiments from scratch using the Adamant user interface or provide an existing schema. At its current state, Adamant is mainly used to compile structured experiment metadata in conjunction with a generic electronic lab notebook. In this talk, we will present the current features of Adamant and production-ready RDM workflows involving Adamant and other RDM tools, as well as concepts for future development of Adamant. These concepts include an ontology and knowledge graph integration for a guided acquisition of structured metadata, and visualization of graph data for better browsing and navigation through the stored metadata. Overall, the ultimate goal of Adamant is to make FAIR RDM activities as easy as possible for researchers.

Slot length:

Poster Session / **39**

Research Software Engineering in the Energy Domain as Part of NFDI4Energy

Author: Corinna Seiwerth[1]

Co-authors: Michael Niebisch [1]; Reinhard German [1]; Stephan Alexander Ferenz [2]

[1] *Friedrich-Alexander-Universität Erlangen-Nürnberg*

[2] *Carl von Ossietzky Universität Oldenburg*

Corresponding Author: corinna.seiwerth@fau.de

Energy researchers often use (self-written) software as a starting point to perform research. Also, this software can be the result of research in this domain, like simulation tools. Research software presents an important research artifact in energy research. Therefore, the National Research Data Infrastructure for the Interdisciplinary Energy System Research (NFDI4Energy) addresses the handling of research software across the entire research and transfer cycle within energy system research projects. For this, the FAIR data principles are applied not only to data but also to software. To improve the FAIRness of research software in the energy domain, we focus on three main aspects:

1. Improve the findability of research software through a registry based on a software ontology
2. Improve interoperability and especially reusability of research software through a simulation middleware

3. Provide an ontology-based approach for integrated development of energy system simulation scenarios through reuse of existing software models

We develop an energy simulation software ontology and a model registry, to improve the findability of research software. The ontology provides a structured overview of different modeling approaches and serves as a guide for researchers. Although, we aim to make the ontology as broad as possible, it only sometimes provides detailed depth in all modeling branches. Therefore, we allow experts to add details for their specific areas of expertise. The software registry complements the ontology and includes links to implementations of simulation techniques as well as test cases and other resources.

The simulation of energy systems often requires the interconnection of models from different disciplines. For this, co-simulation allows the combination of existing models and enables a comprehensive simulation. Co-simulations pose two main problems -the interconnection of different components and data exchange in the model, which leads to technical and conceptual challenges. In addition, potential users of such a coupled simulation model might need more knowledge or resources to implement a co-simulation independently. NFDI4Energy aims to provide easy access to simulation middleware that enables different types of co-simulation.

The complexity and diversity of domains and models in energy system simulation scenarios further require an ontological integration of semantics and domain knowledge in the planning, execution, and evaluation process of interdisciplinary energy system simulations. In this context, we develop ontological structures that integrate specialized hardware-in-the-loop (HIL) and laboratory testing in power system simulation scenarios. We base our approach on an information model that formalizes relationships and properties of simulation models and components and includes references to external model and component registries and the domain-specific ontology.

Overall, NFDI4Energy focuses on multiple supporting activities surrounding reproducible research and best practices in energy system modeling and simulation. This includes a high focus on the (re)use of research software. We would like to present these different aspects regarding research software within NFDI4Energy as a poster. We think that this overview fits perfectly to the scope of the deRSE conference 2024.

Slot length:

Metadata for Research Software / **40**

The SPARQL Unicorn: A Research Tool for Linked Open Data in QGIS and git-action-based ontology documentation

Authors: Florian Thiery[1]; Timo Homburg[2]

[1] *Leibniz-Zentrum für Archäologie (LEIZA)*

[2] *Hochschule Mainz*

Corresponding Authors: timo.homburg@hs-mainz.de, florian.thiery@leiza.de

Introduction

Publishing sustainable research data and providing appropriate access for many research communities challenges many players: Researchers, RSEs, standardisation organizations and data repositories. With national research data infrastructures (NFDI) being set up in Germany, the latter could be solved in the mid- to long-term for specific datasets. In the meantime, researchers often produce datasets of data in research projects which are provided as services, e.g. from a web page, but may, due to a lack of funding, disappear in that form after the research project has ended. To circumvent this, open research data is hosted long-term on public platforms like university libraries, Zenodo or Github. However, this hosted data is not necessarily easily discoverable by different research communities. On top of that, research data is rarely published in isolation, but with links to related

datasets, leading to the creation of link-preserving, FAIR linked open data (LOD) as RDF dumps, modelling data interoperably in common vocabularies. LOD in RDF preserves links, but is not necessarily Linked Open Usable Data (LOUD), i.e. it does not provide data in ways different research communities expect. We would like to address this problem of missing LOUD data while removing requirements on the backend such as hardware and software to a minimum.

Documentation-Tool

We believe that a solution to this data provision problem is publishing research data as static webpages and using standardised static APIs to serve data in ways different research communities expect.

We developed a documentation extension to our SPARQLing Unicorn QGIS Plugin, allowing to publish RDF data dumps as HTML page and RDF serialization per data instance, similar to what frontends to triple stores such as Pubby provide.
It is published as a QGIS Plugin, a standalone script on Github and a Github Action.

The resulting data dump is hostable on static webspaces e.g. Github pages and allows navigating the contents of the LOD data in HTML including a class tree. It may include:
* Further data formats: Graph Data (GraphML, GEXF), General Purpose (CSV)
* SPARQL querying in JavaScript using the data dump
* Generation of static APIs, e.g. JSON documents mimicking standardized APIs, for
* OGC API Features: Access to FeatureCollections from e.g. QGIS
* IIIF Presentation API 3.0: IIIF Manifest Files for images/media in the knowledge graph including typed collections
* CKAN API: Datasets in the DCAT vocabulary or data collections

Static APIs further the accessibility of LOD data for different research communities and increase the chances of data reusage and exposure in different research fields, while at the same time not depending on additional infrastructures for data provision.

Limitations and Future Work

Our talk shows the feasibility of using publicly available examples for geodata and CKAN (SPP Dataset, AncientPorts Dataset, CIGS Datatset) and the ARS-LOD dataset for static IIIF-data.
We discuss requirements and limitations of this kind of publishing in a RDM publishing workflow, in relation to NFDI plans and how to extend this approach to only partially open data using a Solid pod publishing workflow.

Slot length:

Reproducible Packaging / 41

EESSI-going scientific software installations

Author: Alexander Puck Neuwirth[1]

[1] *ITP Universität Münster*

Corresponding Author: alexander.neuwirth@uni-muenster.de

The European Environment for Scientific Software Installations (EESSI, pronounced as “easy”) is a collaboration between different European HPC sites and industry partners, with the common goal to set up a shared stack of optimized scientific software installations.
It can be used on a variety of systems, regardless of the operating system, processor architecture, or GPU of the client system, or whether it’s a full-size HPC cluster, an HTC system, a cloud environment, or a personal workstation, while providing improved reproducibility in scientific computing.
Leveraging the tools and expertise inherent in the HPC community, EESSI integrates solutions such

as EasyBuild, Lmod, CernVM-FS and Gentoo Prefix to ensure the creation of a scalable and rigorously tested software stack that is easy to deploy and maintain.
EESSI also enables users to access the same software stack on their own desktop or laptop, allowing them to run small tests and scale up to larger systems as needed, with a consistent user experience. This presentation provides an exploration of the motivation, design, implementation, and benefits of EESSI.

Docs: http://www.eessi.io/docs
Intro: https://www.youtube.com/watch?v=Fzv4ieiI1jo

Slot length:

Analysis and Visualization / **42**

Handling different analysis workflows in a modular framework

Author: Malte Storm[1]

[1] *Helmholtz-Zentrum Hereon*

Corresponding Author: malte.storm@hereon.de

Helmholtz-Zentrum Hereon operates multiple X-ray diffraction (XRD) experiments for external users and while the experiments are very similar, their analysis is not. Pydidas [1, 2] is a software package developed for the batch analysis of X-ray diffraction data. It is published as open source and intended to be widely reusable.

Because the wide range of scientific questions tackled with the technique of XRD, a limited number of generic tools will not be sufficient to allow all possible analysis workflows. Easy extensibility of the core analysis routines is a key requirement. A framework for creating plugin-based workflows was developed and integrated in the pydidas software package to accommodate different analytical workflows in one software tool. We present the architecture of the pydidas workflows and plugins along with the tools for creating workflows and editing plugins.

Plugins are fairly simple in design to allow users/collaborators to extend the standard pydidas plugin library with tailor-made solutions for their analysis requirements. Access to plugins is handled through a registry which automatically finds plugins in specified locations to allow for easy integration of custom plugins. Pydidas also includes (graphical) tools for creating and modifying workflows and for configuring plugins, as well as for running the resulting workflows.

While pydidas was develop with the analysis of X-ray diffraction data in mind and the existing generic analysis plugins reflect this field, the architecture itself is very versatile and can easily be re-used for different research techniques.

1 https://pydidas.hereon.de
2 https://github.com/hereon-GEMS/pydidas

Slot length:

43

Building a community around your Open Source research software

Authors: Jan Philipp Dietrich[1]; Lavinia Baumstark[None]

[1] *Potsdam Institute for Climate Impact Research (PIK)*

Corresponding Authors: dietrich@pik-potsdam.de, lavinia@pik-potsdam.de

Are you developing an Open Source research software? Is your software developed and maintained mainly by one group or organization? Is your software also of interest to third-party users, be it researchers, NGOs or other group of users? Do you have first third-party users of your software? Do you wonder how to build a community around your software beyond the original developers?
If these questions resonate with you, we invite you to join our upcoming meet-up. This event aims to connect research software developers planning to broaden their development team, those encountering their initial interactions with third-party users, and those who have successfully established vibrant communities around their research software. We are interested in your experiences, both positive as well as negative, in collaborating with software developers beyond your organization and want to learn about the challenges you may have faced.
During the meet-up we will split up into smaller groups and discuss and hopefully answer questions like:
• How to prepare research software for third-party users/developers?
• How to attract new third-party users/developers?
• How to get third-party users into collaborative development?
• How to balance growing demand for support with obligations for own projects?
• How to lower entry barriers?
Your insights will contribute to the creation of a final document summarizing all lessons learned and proposing effective solution strategies. This collective resource will serve as a valuable guide for research software developers navigating the transition from a software primarily used by a specific group to a research software developed and maintained by a broader and more diverse community.

Slot length:

44

Software metadata extraction for Software Management Plans

Authors: Daniel Garijo[1]; Leyla Jael Castro[2]; Marek Suchánek[3]; Stephan Ferenz[4]

[1] *Ontology Engineering group at Polytechnic University of Madrid*

[2] *ZB MED Information Centre for Life Sciences*

[3] *Department of Software Engineering at Czech Technical University in Prague*

[4] *University Oldenburg*

Corresponding Authors: daniel.garijo@upm.es, ljgarcia@zbmed.de, stephan.ferenz@uni-oldenburg.de

Software metadata extraction for Software Management Plans

A hands-on workshop session at DE-RSE 2024

Software Management Plans (SMPs) help Research Software Engineers (researchers who develop code as part of their research or software engineers who support research activities) to oversee some of the activities during the software development lifecycle. Such activities could support (i) researchers in developing better software by following some minimum good practices, and (ii) software engineers in adopting some practices that might not be common outside research (e.g., archiving releases, providing citation information).

In this workshop, we will introduce (research) software metadata and its connection to SMPs, a tool for metadata extraction from GitHub repositories (GitHub, paper, website), and the Software Management Wizard (SMWizard –a tool to facilitate filling in ELIXIR SMPS, see preprint and web page). After the introduction, we will have a hands-on session to work in small groups to try and improve the SMW and the metadata extractor (e.g., suggesting improvements to the SMP/SMW/metadata extraction, using the tools to improve your own GitHub repo machine-readability, implementing a

new data integrator for the Wizard, improving the metadata extraction). The hands-on will finish with feedback from participants, followed by a wrap-up from the workshop organizers.

Organizers

- Leyla Jael Castro. ZB MED Information Centre for Life Sciences. ORCID:0000-0003-3986-0510
- Stephan Ferenz. Department of Computer Science at University Oldenburg. ORCID:0000-0001-9523-7227
- Daniel Garijo. Ontology Engineering group at Polytechnic University of Madrid. ORCID:0000-0003-0454-7145
- Marek Suchánek. Department of Software Engineering at Czech Technical University in Prague.ORCID:0000-0001-7525-9218

Tentative agenda

Time	Activity	Responsible
—	—	—
10’	Welcome	Leyla Jael Castro
Introductory session		
10’	Software metadata	Stephan Ferenz
10’	Machine-actionable SMPs	Leyla Jael Castro
15’	Software metadata extraction	Daniel Garijo
15’	SMWizard	Marek Suchánek
Hands-on session		
90’	In groups, work on one of the following topics: (i) Create your SMP with the SMWizard and brainstorm on improvements, (ii) create an integrator for the SMWizard, (iii) use the metadata extraction tools to produce Codemeta and/or Bioschemas metadata files and brainstorm on improvements, (iv) extend or develop new functionality for the metadata extractor, (v) your own idea	All participants
20’	Feedback from groups	One participant per group
10’	Wrap-up	Leyla Jael Castro

Slot length:

Workshop (3h)

RSE Research / 45

Do you know what researchers do and what they want? Experiences from a survey and user interviews

Authors: Bernadette Fritzsch[None]; Miguel Andrés-Martínez[1]

[1] *AWI*

Corresponding Authors: miguel.andres-martinez@awi.de, bernadette.fritzsch@awi.de

Since the SSI surveys, we have known that research software is an essential part of the scientific work of many researchers. However, many researchers who develop software have not received specific training for that task, with the consequent impacts on the software quality, re-usability and sustainability. Central support units at the institutions where these scientists are employed can be important instruments to overcome this problem.
Such a group has existed at the Alfred Wegener Institute, Helmholtz Center for Polar and Marine Research for several years. It was originally initiated to integrate new numerical methods into various Earth system models. However, it quickly became clear that the need for support goes much further. In addition to specific tasks such as porting model code for new HPC platforms or new programming paradigms, it also includes more general tasks such as assistance in introducing better coding and developing practices and providing training.

Due to technical developments on the one hand and the strong staff fluctuations due to fixed-term contracts in the institute on the other, the range of services has to be constantly adjusted. Therefore, we conducted a survey at our institution to determine the status quo regarding the development of research software. It was found that scientific groups at AWI often invest enormous time and human resources in development and maintenance work. At the same time, the need for support services and consulting was articulated. In addition, our HPC and Data Processing support group organized interviews with specific user groups to better structure the support services of our group, using the small group's staff more efficiently. As a support unit for different working groups, it is important to find out early about planned new scientific projects that require support. At the same time, the interviews served to discover synergies where generalized solutions can support several scientific groups at the same time. Training and workshops for development best practices, code optimization, automation of tasks, etc are of general interest.
Our contribution will present the findings from the survey and interviews, and will introduce the measures taken by our group to help our users to develop code more sustainably.

Slot length:

Continuous Integration - Advanced / 46

Continuous Benchmarking for a Massively Parallel Multi Physics Framework

Authors: Christoph Alt[1]; Harald Köstler[1]

[1] *Friedrich-Alexander Universität Erlangen-Nürnberg*

Corresponding Author: christoph.alt@fau.de

waLBerla, an HPC software framework for multi-physics simulations based on the Lattice Boltzmann method, has consistently demonstrated exceptional parallel performance across various supercomputing platforms.

Maintaining a resilient codebase developed over a decade by multiple generations of developers is a paramount goal for waLBerla.
To achieve this, the framework has employed a continuous integration pipeline for a long time.
This pipeline ensures functional correctness and compatibility with a diverse array of compilers through systematic and automated testing.

Recognizing the importance of detecting performance regressions introduced by code changes, waLBerla has extended its continuous integration setup to include a continuous benchmarking pipeline.
Performance benchmarks run automatically on a range of CPU and GPU architectures, providing developers with swift feedback on how new commits impact the framework's performance.

In adapting to the dynamic landscape of HPC, waLBerla emphasizes versatility by testing on diverse hardware.
This proactive approach not only ensures good performance but also provides developers with quick insights into the effects of their contributions.

Adhering to FAIR principles (Findable, Accessible, Interoperable, and Reusable), waLBerla stores profiling and timing data.
Developers benefit from an interactive visualization of this data, enabling them to discern performance trends over time.
This transparent approach empowers developers to make informed decisions within a performance-driven development process.

In summary, this presentation offers an in-depth exploration of waLBerla's comprehensive development infrastructure.
The delicate balance between functional correctness and performance optimization is achieved through a meticulous Continuous Integration Pipeline, a versatile Continuous Benchmarking Setup, and transparent, FAIR-compliant profiling and timing data visualization.

The audience will gain insights into how these elements collectively cultivate a development environment where waLBerla excels across diverse HPC landscapes.

Slot length:

Metadata for Research Software / **47**

How to "unHIDE"and improve the metadata landscape of research software in Helmholtz.

Author: Jens Bröder[1]

Co-authors: Fiona D'Mello [2]; Gabriel Preuß [3]; Said Fathalla ; Pier Buttigieg [4]; Oonagh Mannix [5]; Volker Hofmann ; Stefan Sandfeld [2]

[1] *Forschungszentrum Jülich GmbH (IAS-9)*

[2] *Forschungszentrum Jülich*

[3] *HZB*

[4] *GEOMAR Helmholtz Centre for Ocean Research*

[5] *HMC matter/HZB*

Corresponding Author: j.broeder@fz-juelich.de

Research across the Helmholtz Association is based on inter- and multidisciplinary collaborations across its 18 centers and beyond. However, the wealth of Helmholtz's (meta)data and digital assets are stored in a distributed and incoherent manner, with varying quality.

To address this challenge, the Helmholtz Metadata Collaboration (HMC) launched the unified Helmholtz Information and Data Exchange (unHIDE) project in 2022. UnHIDE aggregates metadata harvested from Helmholtz infrastructure in the Helmholtz Knowledge Graph (Helmholtz KG). This serves as a lightweight and sustainable interoperability layer to interlink data infrastructures and increase visibility and access to the Helmholtz Association's (meta)data and information assets
Version 1.0.0 of the Helmholtz KG was released in October 2023. This includes a comprehensive web front end for manual search of resources 1, a stable and documented 2 backend with a tested data ingestion and integration pipeline, and machine accessible endpoints 3.

In this talk we present an overview of the Helmholtz metadata ecosystem, we describe the semantic and technological architecture of the Helmholtz KG and how it integrates metadata from heterogeneous sources to improve visibility and findabiltiy. We will show how code and research software is scattered throughout different platforms (such as institutional gitlab instances), how its metadata is lacking connection to other (research) publications and that only a minority is formally published in central indexes 4. We will show and discuss some results of our efforts to integrate and improve software metadata in Helmholtz as well as future ways how the Helmholtz KG is envisioned to harmonize and improve quality of metadata at the source: in the respective infrastructures.

1 https://search.unhide.helmholtz-metadaten.de/
2 https://docs.unhide.helmholtz-metadaten.de/
3 https://sparql.unhide.helmholtz-metadaten.de/
4 e.g. https://helmholtz.software/

Slot length:

Poster Session / **48**

FID Physik –updating the information infrastructure for physics

Author: Holger Israel[1]

Co-author: Julia Hoffmann [1]

[1] *TIB - Leibniz-Informationszentrum Technik und Naturwissenschaften*

Corresponding Author: holger.israel@tib.eu

Share your thoughts on how research software metadata can power FAIR and open science!

In a world where the published scientific literature is growing exponentially, researchers looking for specific scientific information need to retrieve their signal from a rising flood of information noise. The adoption of semantic technologies such as knowledge graphs can mitigate this issue –while at the same time help building a more FAI and open research environment.

High-quality metadata, including those describing research software are crucial building blocks for semantic solutions as well as for AI applications for scientific libraries. Currently, however, these metadata are often insufficient in quantity and quality. Physics research in particular would benefit from improved and more standardised annotation practices.

Hence, TIB –Leibniz Information Centre for Science and Technology, together with partners at Physikalisch-Technische Bundesanstalt (PTB) and Leibniz Institute for Plasma Science and Technology (INP) are going to propose a "Fachinformationsdienst (FID) Physik", a specialised information service for physics. This support infrastructure aims to improve researchers'access to research-specific information and to provide services based on the aggregation and curation of high quality physics metadata.

This is where we need your help –and would like to offer our help for you, too!

Research software engineering plays a crucial role in the production, curation, maintenance, and distribution of research outputs in the physical sciences. Thus, research software engineers are key stakeholders for shaping the transformation towards Open Science and FAIR physics research data.

Visit our interactive poster and let us know which of our ideas regarding research software metadata we should prioritise. What should we add to support your work as a research software engineer who is involved in physics research? We would like to incorporate your suggestions and comments as a part of our community survey into our proposal for the FID Physik.

Slot length:

Parallelization and HPC Infrastructure / 49

SPHinXsys Parallelization with SYCL: Smoothed Particle Hydrodynamics on Heterogeneous Systems

Author: Alberto Guarnieri[1]

Co-author: Xiangyu Hu

[1] *Technical University of Munich*

Corresponding Author: xiangyu.hu@tum.de

Simulations based on particle methods, such as Smoothed Particle Hydrodynamics (SPH), are known to be computationally demanding, consisting on numerous interactions between locally-defined neighbors. Compared to other numerical methods, SPH is mesh-free, meaning that computations are not restrained to a fixed grid: particles, acting as interpolation nodes, are instead free to move across the entire domain, leading to additional challenges when dealing with parallel computations. While such methods have for long been executed in parallel on multi-core CPUs, in recent years the increasing adoption of many-core accelerators, such as GPUs, has opened up the field of parallel computing to new possibilities.

However, parallel models and techniques do often differ between multi-core and many-core systems, requiring particular attention in coordinating the execution of threads and memory operations for the latter.
Moreover, hardware fragmentation and vendor-specific programming interfaces are still characterizing their market. Hence, support for various hardware configurations may easily lead to non-trivial and less maintainable implementations.
To leverage over those differences, some higher-level specifications have become available recently, such as the SYCL programming standard, which provides an interface for compiling ISO C++ code on various back-ends, including GPU APIs.
The following work highlights the initial effort in adopting the SYCL standard for the execution of SPHinXsys, an open-source multi-physics library. The result is an execution model able to run the same implementation on variable (heterogeneous) hardware, with considerable speed-up compared to the current multi-core CPU parallelization.
The discussion will primarily focus on the difference between multi-core and many-core parallelization, describing how the existing parallel methods have been adapted to be executable with SYCL. Among others, representation of data-structures for parallel access, communication strategies, and parallel methods for data sorting will be topics discussed in depth. Minimizing the effort for the user to adopt this new execution model has also been taken into consideration, reducing the changes required to port an existing simulation. Execution details are designed to be transparent to the library user, not requiring particular knowledge of the underlying execution model. Finally, benchmarks will be presented, showcasing performance comparisons between the current multi-core CPU implementation and the newly introduced SYCL parallelization with a GPU back-end.

Slot length:

50

FAIR and Reproducible Code

Authors: Heidi Seibold[1]; Tobias Schlauch[2]

[1] *Digital Research Academy*

[2] *DLR / HIFIS*

Corresponding Authors: tobias.schlauch@dlr.de, inbox@heidiseibold.de

This workshop is organised by the German Reproducibility Network.

<h3 id="title-fair-and-reproducible-code">Title: FAIR and Reproducible Code</h3>

<h3 id="abstract">Abstract</h3>

Let's talk about good practices in research as well as research software engineering! We love the saying "Better Software, Better Research", but what we actually need to do in practice? In this workshop we share best practices on how to make your work FAIR and reproducible and then also discuss what that means for your project. How can I set up a reproducible project? How should I license my software? How can I make my materials findable and reusable? These and more questions will (hopefully) be answered as part of this workshop.

<h3 id="goals">Goals</h3>

 Spotlight on the German Reproducibility Network Provide overview about current good practices Provide hands-on experience and implement first ideas

<h3 id="schedule">Schedule</h3>

<table> <thead> <tr class="header"> <th>Topic</th> <th>Details</th> <th>Speaker</th> <th>Duration (min)</th> <th>Start</th> </tr> </thead> <tbody> <tr class="odd"> <td>Arrival + Welcome</td> <td>Welcome (from GRN+HIFIS)</td> <td>Tobias</td> <td>5</td> <td>09:30</td> </tr> <tr class="even"> <td>Introduction</td> <td>The GRN and the idea of the workshop</td> <td>Heidi</td> <td>10</td> <td>09:35</td> </tr> <tr class="odd"> <td>Talk session</td> <td>Short talks introducing the 4 topics (5min + questions)</td> <td>Heidi, Michael, Tobias, Max</td> <td>40</td> <td>09:45</td> </tr>

<tr class="even"> <td>Break</td> <td>—</td> <td>—</td> <td>20</td> <td>10:25</td> </tr> <tr class="odd"> <td>Intro: World Cafe</td> <td>How the world cafe works</td> <td>Heidi</td> <td>5</td> <td>10:45</td> </tr> <tr class="even"> <td>World Cafe</td> <td>Participants can go to tables with experts on the 4 topics (15 minutes on each table)</td> <td>all</td> <td>65</td> <td>10:50</td> </tr> <tr class="odd"> <td>Wrap-Up</td> <td>Wrap-up (1 minute highlight per table)</td> <td>Tobias (+ Heidi, Max, Michael)</td> <td>5</td> <td>11:55</td> </tr> </tbody> </table>

<h3 id="topics">Topics</h3>

4 topics: 4 talks + 4 world cafe tables (15min per round / per table)

<h4 id="setting-up-a-fair-and-reproducible-project-repository">Setting up a FAIR and reproducible project repository</h4>

 The role of a README Folder structure and good naming for reproducibility and seamless collaboration Making contributions possible ⋯

 Expert: Heidi

<h4 id="stabilizing-the-computing-environment">Stabilizing the computing environment</h4>

 Docker, VM, ⋯ Different approaches to locking dependencies and/or environment VM vs. container images vs. *.lock files

 Expert: Michael

<h4 id="software-and-data-licenses">Software and data licenses</h4>

 Licenses as a integral component to make your software and data FAIR and reusable

 Expert: Tobias

<h4 id="reproducible-communication">Reproducible communication</h4>

 How to communicate research outputs in a reproducible and FAIR way? Quarto, Shiny co.

 Expert: Max

Slot length:

other(help with comment)

Lessons learned and applied / 51

Developing the ClusterCockpit Monitoring Framework - An Odyssey from PHP to Go

Author: Jan Eitzinger[1]

[1] *NHR@FAU, Friedrich-Alexander Universität Erlangen-Nürnberg*

Corresponding Author: jan.eitzinger@fau.de

ClusterCockpit, a specialized performance and energy monitoring framework designed for High-Performance Computing (HPC) cluster systems, has evolved significantly since its inception in 2018. The framework comprises a web frontend, an API backend, a node agent, and a metric in-memory cache. Being an open-source project, its source code is available on GitHub at https://github.com/ClusterCockpit.

This presentation delves into the challenges encountered and the journey taken by ClusterCockpit over the past five years. Initially built as a PHP Symfony web application relying on server-side rendering and JQuery libraries, the framework has transformed into its current state with a Go API backend and a web frontend based on Svelte.

The talk emphasizes the tradeoffs encountered when choosing frameworks at different levels and finding the right balance between ease of use and flexibility. The project's progression is explored, starting from its early stages with PHP Symfony to the current architecture. Notable stages and experiences are highlighted, providing insights into the decision-making process.

Particular attention is given to the choices made in terms of architecture and design, shedding light on the considerations that led to the adoption of Go for the backend and Svelte for the frontend. The presentation aims to offer a comprehensive understanding of ClusterCockpit's development, focusing on the evolution of technologies, frameworks, and the project's current state.

Slot length:

Continuous Integration - Intermediate / **52**

Testing - to Unit Tests and Beyond

Author: Jakob Fritz[1]

Co-author: Robert Speck [1]

[1] *Forschungszentrum Jülich GmbH*

Corresponding Author: j.fritz@fz-juelich.de

In this presentation, we will dive into the topic of testing, with a specific focus on the development of unit tests. Fundamental approaches to writing effective tests and improving the quality of our software will be explained. We will go into why to do testing at all and how it helps us to detect bugs early and enhance the maintainability of our codebase.

In order to be able to determine how well our code is tested, the concept of code coverage is introduced. Furthermore, an outlook on additional possibilities and approaches for writing tests will be provided. This will include advanced testing concepts such as integration tests, property based testing, and mutation tests.

The objective of this presentation is to provide participants with a basic understanding of unit test development and equip them with practical tips and techniques for building high-quality software. We hope that after this talk, you will be able to write tests effectively and optimize your development processes.

Slot length:

other(help with comment)

RSE Research / **53**

Do Scientists Write About Software? - An RSE Publication Monitor

Author: Jakob Fritz[1]

Co-author: Robert Speck [1]

[1] *Forschungszentrum Jülich GmbH*

Corresponding Author: j.fritz@fz-juelich.de

The RSE Publication Monitor is an online tool to show how much and in which context scientists write about software. As the RSE Publication Monitor is developed at the Forschungszentrum Jülich, it currently uses a publication database of the Forschungszentrum as data source.

The monitor regularly queries a database to find publications that mention certain keywords related to software. Based on this information, it is possible to browse through correlations of keywords mentioned in these publications, and also to find out how often different keywords are used in publications from different institutes/research groups.

This tool can be easily extended to cover other keywords of interest. Furthermore, the code is open source and can be easily adapted to use other data sources and everyone is encouraged to design their own analysis. Furthermore, we are always open for collaborations and further ideas.

This talk will not only focus on the general approach and the technical structure of the publication monitor, but will also show some findings from using the monitor in order to highlight the benefits of setting up such a monitor.

Slot length:

Policies and Legal Aspects / 54

Research Software at Helmholtz - tying up different knots

Author: Christoph Bruch[1]

Co-authors: Antonia C. Schrader [2]; Heinz Pampel [2]; Lea Maria Ferguson [2]; Lena Messerschmidt [2]; Marcel Meistring [2]; Nina Weisweiler [2]; Paul Schultze-Motel [2]; Roland Bertelmann [2]; Steffi Genderjahn [2]

[1] *Helmholtz Open Science Office*

[2] *Helmholtz-Gemeinschaft*

Corresponding Author: christoph.bruch@gfz-potsdam.de

Research Software is one of the most important tools of modern science, and the development of research software is often a prerequisite for cutting-edge research. With the advancing digitalization of research and teaching, the number of software solutions emerging at scientific institutions with the purpose of gaining scientific knowledge is ever increasing. For the reproducibility of scientific results, referencing and making accessible the respectively employed or developed software is essential. In many cases, the provision of corresponding software is of great importance for the reproducibility of data analyses as well as for the re-use of the research data in question. RSE is therefore an integral part of the research process and is addressed as such in research policy at Helmholtz, such as the Helmholtz Open Science Policy and individual Policies at the Helmholtz Centers. For large organizations, coordination of great numbers of activities and integration of a wide variety of stakeholders is a challenge. Furthering der communication among these stakeholders is at the core of the mission of the Helmholtz Open Science Office. This talk will provide an insight of how this is approached at Helmholtz, thereby giving an overview of activities at Helmholtz supporting and recognizing the production and maintenance of research software, such as the Helmholtz Task Group Research Software, Helmholtz Federated IT Services (HIFIS), Helmholtz Research Software Directory, the Helmholtz Incubator Software Award, the Helmholtz Platform for Research Software Engineering - Preparatory Study (HIRSE_PS), licensing workflows, the Helmholtz Open Science Policy, Helmholtz engagement for research reproducibility and activities linking all RSE stakeholder at Helmholtz as well as engaging with the RSE community on a national and international level.

Slot length:

55

RSE in Max-Planck

Author: Stephan Janosch[1]

[1] *MPI-CBG*

Corresponding Author: janosch@mpi-cbg.de

RSEs within Max-Planck would like to meet and discuss internally the current state of affairs.

Most likely we will be less than 30 people. Maybe we will start at 9.30 and go until 12. A break in between.

Slot length:

Workshop (3h)

Poster Session / 56

Research Software Engineering in the NFDI

Authors: Bernd Flemisch[1]; Florian Thiery[2]

[1] *Institute for Modelling Hydraulic and Environmental Systems, University of Stuttgart*

[2] *Leibniz-Zentrum für Archäologie (LEIZA)*

Corresponding Authors: bernd.flemisch@iws.uni-stuttgart.de, florian.thiery@leiza.de

Research Software Engineering is a fundamental part of the German National Research Data Infrastructure (NFDI). In accordance with that, a “deRSE Arbeitskreis NFDI” will serve as a connection point for RSEs in the NFDI inside deRSE e.V.

Within the NFDI e.V., several “sections” are dealing with overarching topics, e.g., the “Sektion Common Infrastructures” with its working groups on “Data Integration (DI)”, “Data Management Planning (DMP)”, “Data Science and Artificial Intelligence (DSAI)”, “Electronic Lab Notebooks (ELN)”, “Persistent Identifiers (PID)” and “Research Software Engineering (RSE)”.

The RSE working group connects the NFDI consortia in software-related aspects. It focuses on three areas: Research software, software communities and software infrastructure at NFDI. In an advisory and supportive capacity, the working group operates a central forum and establishes the necessary software ecosystem within NFDI for the professional development of software infrastructure components, which in their entirety represent an integral part of the NFDI. In addition, the working group serves as an interface for the NFDI to comparable European and international initiatives in order to promote the connectivity of the NFDI with other infrastructures.

This meet-up would like to bring RSEs already working or interested in the NFDI together to create an active network and encourage RSEs to be part of the RSE working group in the NFDI Section Infra.

Slot length:

other(help with comment)

Poster Session / 57

Harvester-Curator, a tool to elevate metadata provision in data and/or software repositories

Author: Sarbani Roy[1]

Co-authors: FangFang Wang [1]; Dennis Gläser [1]

[1] *University of Stuttgart*

Corresponding Author: sarbani.roy@simtech.uni-stuttgart.de

In today's scientific landscape, especially in robust data-driven research, metadata plays a pivotal role. Referred to as "data about data," metadata is essential for augmenting the FAIRness (Findability, Accessibility, Interoperability, and Reproducibility) of digital information. It provides crucial context and structure to raw data, encompassing details such as origin, format, and provenance. Moreover, it contributes to enhancing research reproducibility by documenting methodology, parameters, and conditions, promoting transparency and validation. In data and/or software repositories, metadata acts as the guardian of data integrity and accessibility. A well-curated repository relies on robust metadata for efficient categorization, indexing, and retrieval.

Despite its significance, researchers often face challenges in providing metadata, primarily due to time constraints, a lack of guidelines, and the perceived complexity involved. In many instances, compiling metadata relies heavily on manual procedures. Researchers must gather metadata at critical stages of their research—either concurrently with their ongoing work (which may disrupt the researcher's workflow) or by retrospectively revisiting the entire research process (which is cumbersome). This also demands a comprehensive understanding of potential metadata fields, which may vary from one target repository to another. Identifying the relevance of a piece of information in terms of metadata is challenging, adding an additional layer of complexity and hindering the achievement of comprehensive and varied metadata for publication. Consequently, there is a compelling need for the development and implementation of an automated metadata collection process to simplify this intricate facet of research data management. Effectively addressing these challenges requires the creation of a specialized tool to automate the gathering of contextual information—an essential step in streamlining the complex task of metadata provision.

We are introducing Harvester-Curator, a tool designed to elevate metadata provision in data and/or software repositories. In the first phase, Harvester-Curator acts as a scanner, navigating through user code and/or data repositories to identify suitable parsers for different file types. It collects metadata from each of the files by applying corresponding parsers and then compiles this information into a structured JSON file, providing researchers with a seamless and automated solution for metadata collection. Moving to the second phase, Harvester-Curator transforms into a curator, leveraging the harvested metadata to populate metadata fields in a target repository. By automating this process, it not only relieves researchers of the manual burden but also ensures the accuracy and comprehensiveness of the metadata. Beyond its role in streamlining the intricate task of metadata collection, this tool contributes to the broader objective of elevating data accessibility and interoperability within repositories.

Slot length:

Poster Session / 58

Empowering HPC Workflows with Snakemake: A Comprehensive Training Program

Author: Malte Petersen[1]

Co-authors: Aasish Kumar [2]; Christian Meesters [3]; Fabian Brand [4]; Florian Boecker [4]; Johannes Köster [5]; Lukas Hellmann [6]; Martin Paleico [2]

[1] *University of Bonn, High Performance Computing & Analytics Lab*

[2] *GWDG*

[3] *Johannes Gutenberg Universität Mainz*

[4] *University of Bonn*

[5] *University of Duisburg-Essen*

[6] *Johannes Gutenberg University of Mainz*

Corresponding Author: malte.petersen@uni-bonn.de

In the evolving landscape of High-Performance Computing (HPC), the ability to efficiently design, implement, and manage scientific data analysis workflows is paramount. We recognize this need and present a unique training program tailored for researchers and administrators keen on harnessing the power of the Snakemake workflow system in an HPC environment.

This course material offers a comprehensive curriculum for mastering Snakemake, a versatile workflow management system known for its simplicity, scalability, and reproducibility. Aimed at both beginners and experienced users, our material covers the entire spectrum of HPC conformant workflows, enabling participants to navigate the intricacies of modern computing clusters with ease. We also introduce software engineering best practices in the framework of Snakemake and its workflow and wrapper ecosystem so that creators will be able to contribute their own designs.

The material is designed to be modular and site-agnostic. We invite teachers to adapt the slideset to their requirements. To facilitate this, we offer a flexible configuration system that allows to specify site-specific information in simple configuration files.f

Slot length:

Parallelization and HPC Infrastructure / 59

The Helmholtz Analytics Toolkit (Heat) and its role in the landscape of massively-parallel scientific Python

Authors: Claudia Comito[1]; Markus Götz[2]; Juan Pedro Gutirrez Hermosillo Muriedas[3]; Björn Hagemeier[4]; Fabian Hoppe[5]; Philipp Knechtges[6]; Kai Krajsek[4]; Alexander Ruettgers[7]; Michael Tarnawa[8]

[1] *Forschungszentrum Jülich, Jülich Supercomputing Centre*

[2] *Karlsruhe Institute of Technology*

[3] *SCC*

[4] *Forschungszentrum Jülich GmbH*

[5] *DLR - Institut für Softwaretechnologie - HPC*

[6] *DLR*

[7] *German Aerospace Center (DLR)*

[8] *Forschungszentrum Jülich*

Corresponding Author: fabian.hoppe@dlr.de

When it comes to enhancing exploitation of massive data, machine learning methods are at the forefront of researchers'awareness. Much less so is the need for, and the complexity of, applying these techniques efficiently across large-scale, memory-distributed data volumes. In fact, these aspects typical for the handling of massive data sets pose major challenges to the vast majority of research communities, in particular to those without a background in high-performance computing. Often,

the standard approach involves breaking up and analyzing data in smaller chunks; this can be inefficient and prone to errors, and sometimes it might be inappropriate at all because the context of the overall data set can get lost.

The Helmholtz Analytics Toolkit (Heat) library offers a solution to this problem by providing memory-distributed and hardware-accelerated array manipulation, data analytics, and machine learning algorithms in Python. The main objective is to make memory-intensive data analysis possible across various fields of research —in particular for domain scientists being non-experts in traditional high-performance computing who nevertheless need to tackle data analytics problems going beyond the capabilities of a single workstation. The development of this interdisciplinary, general-purpose, and open-source scientific Python library started in 2018 and is based on collaboration of three institutions (German Aerospace Center DLR, Forschungszentrum Jülich FZJ, Karlsruhe Institute of Technology KIT) of the Helmholtz Association. The pillars of its development are…

- to enable memory distribution of n-dimensional arrays,
- to adopt PyTorch as process-local compute engine (hence supporting GPU-acceleration),
- to provide memory-distributed (i.e., multi-node, multi-GPU) array operations and algorithms, optimizing asynchronous MPI-communication (based on mpi4py) under the hood, and
- to wrap functionalities in NumPy- or scikit-learn-like API to achieve porting of existing applications with minimal changes and to enable the usage by non-experts in HPC.

In this talk we will give an overview on the current state of our work. Moreover, focussing on the research software engineering perspective we will particularly address Heats role in the existing ecosystem of distributed computing in Python as well as technical and operational challenges in its further development.

Slot length:

Policies and Legal Aspects / 60

RSE and the aspects of communication and education –a tale from the shores of scientific coding

Author: Benjamin Fuchs[1]

[1] *Deutsches Zentrum für Luft- und Raumfahrt e.V.*

Corresponding Author: benjamin.fuchs@dlr.de

Communication and teamwork are important parts of modern research software engineering in a multi-disciplinary field.

In our work we are based on a well-established foundation of trusted tools from industry and open source communities like git, continuous integration, test frameworks or issue tracker and KANBAN boards for shaping and helping our workflows. While research software engineering frequently means transferring knowledge, be it with our peers, our students, or our clients, we also are challenged in unique ways to communicate and educate.

The talk gives an overview over how we at the DLR Institute of Networked Energy Systems in the Department of Energy Systems Analysis have adapted methods from a variety of fields. Including our experiences:

- in adapting SCRUM and what did or did not work
- with retrospectives especially how, these can be applied in an education setting
- how non-violent communication techniques can be beneficial overall and specifically with requirements engineering

- how clear scopes and roles help us keep software projects on track
- with effort estimation and why one should train it early on

The talk ends with an overview over how each of the mentioned pieces work together to reduce the complexity in and around our work and to free resources for writing code and contributing to science.

Slot length:

61

Introduction to ioProc, a scientific workflow manager for RSE

Authors: Benjamin Fuchs[1]; Felix Nitsch[1]; Jan Buschmann[1]

[1] *Deutsches Zentrum für Luft- und Raumfahrt e.V.*

Corresponding Authors: jan.buschmann@dlr.de, benjamin.fuchs@dlr.de

ioProc is a scientific workflow manager designed for reproducible, maintainable, and transparent linear workflows which are metadata ready. In this tutorial we will together:

- Install ioProc with pip into a virtual environment and briefly explain the dependencies
- Setup a new workflow with the integrated quickstart command line tool and look at an alternative approach by setting it up manually
- Execute workflows from the command line
- Build a simple workflow from scratch including the caching feature for rapid development
- Follow the FAIR principles by reading and storing metadata after data processing
- Integrate another person's workflow into our workflow
- And last but not least showcase how one can transition from an ioProc workflow to a standalone application with reduced overhead.

Finally we will close the tutorial with a quick guided tour through the source code repository of ioProc which is particularly interesting for potential contributers or for the ones who want to peek under the hood.

What do you need to bring with you?

- A working conda or mambaforge installation or virtual environment software of your choice
- The downloaded example data that we will publish in advance before the tutorial
- A mood of exploration

Slot length:

Workshop (3h)

Cross-Platform Development with C/C++ / 62

Cross-platform deployment of a complex C++ computational software with GUI and Python API

Author: Ammar Nejati[1]

Co-authors: Joachim Wuttke [2]; Mikhail Svechnikov [2]

[1] *Forschungszentrum Jülich / Jülich Centre for Neutron Science JCNS at Heinz Maier-Leibnitz Zentrum (MLZ), Forschungszentrum Jülich GmbH*

[2] *FZJ/JCNS-4*

Corresponding Author: a.nejati@fz-juelich.de

Deployment of scientific software across diverse platforms like MacOS, Windows, and Linux is a requirement for any software which is developed for the broader community. Such deployment presents multifaceted challenges, particularly when mixed-language programming (e.g., C/C++, Fortran and Python) with intricate dependencies (like Qt or Python) is a part of the build mechanism.

Addressing library dependencies adds another layer of complexity, particularly in the choice between static or dynamic linking and maintaining consistency across versions and platforms. Furthermore, bundling libraries with the installer/package requires strategic choices to make a balance between efficient use of system resources, portability and easiness of installation for the normal user (with Windows MSI installers, MacOS DMG files, and Linux packages like DEB and RPM).

The incorporation of a separate Python package ("wheel") provides an straightforward installation mechanism for the user as Python's cross-platform compatibility simplifies certain deployment aspects and usage, yet incorporating the underlying C/C++ components (libraries) necessitates a proper configuration to ensure a smooth integration within the user's system.

This talk explores the complexities involved in deploying such software, with attention to platform-specific nuances, intricacies in terms of library linking, compilation and packaging (installers and Python wheels), and provides some insights and solutions acquired in the long-term experience with developing BornAgain, an open-source software to simulate and fit neutron and x-ray scattering.

Slot length:

Teaching RSE Skills / 63

Experiences from first time RSE class at the Computer Science Faculty of TU Dresden

Author: Guido Juckeland[1]

[1] *Helmholtz-Zentrum Dresden-Rossendorf*

Corresponding Author: g.juckeland@hzdr.de

For the first time, Introduction to Research Software Engineering was offered as a class in the Computer Science faculty of TU Dresden during this winter semester (2023/24) (https://tu-dresden.de/ing/informatik/smt/cg
This talk will briefly cover the content, feedback from students and own observations as well as ideas how to continue and extend the class in the future.

Slot length:

Poster Session / 64

kdotpy: Calculating the electronic properties of topological insulators using k·p theory

Author: Wouter Beugeling[1]

Co-authors: Maximilian Hofer [1]; Christian Berger [1]; Florian Bayer [1]; Tobias Kießling [1]

[1] *Julius-Maximilians-Universität Würzburg*

Corresponding Author: wouter.beugeling@physik.uni-wuerzburg.de

The exotic properties of topological insulators were theoretically predicted and experimentally realized around 2005. While for some parameters (material composition, dimensions of the device) it is possible to use simplified, analytically solvable models to describe the physics, this is not generally the case.

In order to simulate the devices fabricated and measured at the Institute for Topological Insulators and Experimentelle Physik III in Würzburg, we use band structure calculations based on k·p theory. In its simplest form, the problem boils down to diagonalization (finding the eigenvalues and eigenvectors) of an 8×8 matrix that depends on momentum variables.

However, the geometry of the devices, typically a layer stack of several different materials, dictates spatial confinement in one or two dimensions. In the confined dimensions, the momentum has to be converted to a discretized lattice of real-space coordinates. The matrix size grows to $\sim 10^3 \times 10^3$ for one and $\sim 10^5 \times 10^5$ for two confinement directions. Diagonalization of such large matrices is a computationally demanding problem that requires sparse algorithms to be solved.

Our Python application kdotpy provides the infrastructure to tackle these problems. The core of kdotpy is the construction of the large Hamiltonian matrices in a sparse format and the application of a suitable diagonalization solver. The user can choose the solver, depending on the nature of the problem and the available hardware: The default solver in kdotpy is the sparse solver 'eigsh' from SciPy (based on ARPACK), but kdotpy also provides an interface to GPU accelerated solvers using CuPy. Very large problems that require large amounts of memory (about 10 GB per data point and above) can be run on the HPC cluster. If needed, the problem can be split up into smaller ones, and the data put together afterwards using a tool included with kdotpy.

Moreover, kdotpy provides functions to "post-process" the diagonalization results and convert them to a form that allows for a direct comparison with the measurements on the devices. In particular, we provide a conversion between energy and carrier-density dependence, relevant for magnetotransport experiments for example, and we calculate optical transitions for comparison to spectroscopic measurements. The output is provided in the form of images, data files in CSV format, and an XML-based serialized data format designed to be maximally compatible between versions and to be suitable for long-term data storage.

Slot length:

RSE Communities / 65

Improving research through training - the Digital Research Academy

Author: Heidi Seibold[1]

[1] *Digital Research Academy*

Corresponding Author: inbox@heidiseibold.de

When we get into science, we do so because we want to have an impact on the world, explore the boundaries of knowledge, or just answer questions. Yet, we see so much bad quality science, results that are not reproducible, and building on the shoulders of giants seems to be so far away.
One building block of research quality is bringing the right knowledge to the right people. Open Science is hard. Data analyses are hard. Research Software Engineering is hard. So how do we get these skills across? Of course, through training.

In this talk I want to present a new trainer network: the Digital Research Academy. Since research software engineering is one of our core topics, I want to share my thoughts on educating the next generation of researchers and RSEs, the way the Digital Research Academy can contribute to it, as well as encourage current RSEs to become ambassadors for digital literacy.

Slot length:

RSE Communities / **66**

RSE community building activities in Switzerland: recent activities and future plans

Authors: Franziska Oschmann[1]; Tarun Chadha[1]; Uwe Schmitt[1]

[1] *Scientific IT Services ETH ZUrich*

Corresponding Author: uwe.schmitt@id.ethz.ch

We are data scientists and software engineers from ETH Zurich's Scientific IT Services, an embedded RSE and HPC group.

After visiting RSE con UK for the second time, we started to build up an RSE community at ETH Zurich and to connect interested people within Switzerland. The response so far has been overwhelming and we want to give a short overview of what we have done up to now and what we are planning for the future. We also hope to connect with RSE communities outside of Switzerland for future collaborations.

Slot length:

67

Community Feedback session for de-RSE position paper "Establishing RSE departments in German research institutions"

Authors: Dominic Kempf[1]; Frank Löffler[2]

[1] *Scientific Software Center, Heidelberg University*

[2] *Universität Jena*

Corresponding Author: dominic.kempf@iwr.uni-heidelberg.de

A working group of de-RSE e.V. is currently writing a position paper on "Establishing RSE departments in German research institutions". According to the timeline of the working group, a preprint version of this position paper will be available for de-RSE24. The paper is intended for adoption as an official position of the de-RSE association. Following up on successful community interactions at de-RSE23 in Paderborn and the de-RSE Unconference 2023 in Jena, we intend to use the conference to collect feedback for the community approval process of the position paper. After the feedback from the conference meet-up and the open online review is incorporated, we aim for a swift adoption and publication procedure for the position paper.

Slot length:

Workshop (1h)

Parallelization and HPC Infrastructure / 68

Reconciling Life Sciences and HPC

Author: Christian Meesters[1]

Co-authors: Johannes Köster ; Lukas Hellmann [1]

[1] *Johannes Gutenberg Universität Mainz*

Corresponding Author: meesters@uni-mainz.de

Despite the immense computational power offered by HPC clusters(and the resources governments pour into obtaining these resources), it remains uncharted territory for many researchers. The primary deterrents include the perceived bureaucratic hurdles associated with accessing HPC resources, the opacity of usage procedures, and a notable lack of accessible support systems.
The intricate process of gaining access to HPC systems, coupled with the convoluted nature of usage procedures, has led to a situation where life science researchers seek alternatives. Institutions, responding to this demand, resort to establishing additional localized infrastructures, resulting in a fragmented and non-standardized compute ecosystem with lots of redundancy. The consequence is a landscape marked by redundancy, inefficiency, and a lack of standardized practices, all of which hinder the realization of the true potential of HPC in life science research.
To bridge this gap, a pivotal step involves the adoption of contemporary workflow systems that seamlessly integrate with HPC batch systems and support remote file management for research data management support. This talk tells the story of incorporating native batch system support into the snakemake workflow management system, advancements by which life science researchers can overcome traditional obstacles and tap into the full capabilities of distributed cluster computing. Highlighted are success stories from massive pharmaceutical ligands screens and genome oriented research.
As the talk explores these advancements, we extend an invitation to engage with the Snakemake developer community. Collaboratively, we can contribute to making bioinformatic workflow solutions more accessible and aligned with Open Science goals. Join us in this practical journey towards a more efficient and collaborative future in life science research.

Slot length:

Poster Session / 69

Introducing an RSE Course at the University of Potsdam - First-Time Experiences and Plans for the Next Iteration

Authors: Nikolas Bertrand[1]; Akshay Devkate[2]; Anna-Lena Lamprecht[1]

[1] *Universität Potsdam, Institut für Informatik und Computational Science*

[2] *University of Potsdam*

Corresponding Authors: anna-lena.lamprecht@uni-potsdam.de, nbertrand@uni-potsdam.de

We introduced a new Research Software Engineering (RSE) course at the University of Potsdam to help students from different backgrounds acquire important RSE skills and supplement their existing programming knowledge. This initiative is in response to the increase of basic programming

skills among students from different fields, but the lack of experience of how to apply them in actual research contexts. The class, which was first offered in the summer term of 2023, consisted of lectures about relevant RSE themes, using the online book "Research Software Engineering with Python" (Irving et al.). Additionally, it featured sections on FAIR software principles, software engineering techniques like requirement engineering and architectural modelling, and computational workflows.

The first iteration of the course surpassed expectations, drawing around 60 participants from 10 study programs, such as computer science, business informatics, physics, computer linguistics, and data science. Attendees exhibited great eagerness and interest in obtaining RSE knowledge. The course layout, which comprised of solitary Jupyter notebook projects (developing a computational narrative around a dataset from the national statistics office) and collaborative group projects (developing a software solution for a research question posed by one of the group members), was very much appreciated.

Following the success of the first iteration, the RSE course at the University of Potsdam intends to increase participation to create a more diverse group. Steps will be taken to tackle formalities and regulations that may obstruct student enrollment. Furthermore, we commit to encouraging cross-disciplinary partnerships by creating more diverse project teams and considering adding a talk on cross-disciplinary and cross-cultural communication. In the next round, we will furthermore put a greater emphasis on workflow automation by integrating tools like Snakemake and customizing the group task to achieve a workflow application.

Slot length:

Continuous Integration - Current Research and Beyond! / 70

Integrated Continuous Benchmarking

Author: Dirk Brömmel[None]

Co-authors: Jakob Fritz [1]; Robert Speck

[1] *FJZ, JSC*

Corresponding Author: d.broemmel@fz-juelich.de

When developing research software, it is often relevant to track its performance over time. It is even vital when targeting high-performance computing (HPC). Changes to the software itself, the used toolchains, or the system setup should not compromise how fast users obtain their results. Ideally, performance or scalability should only ever increase. Hence benchmarking should be an integral part of testing, in particular for HPC codes. At the same time, up-to-date benchmarks that are publicly available can advertise the code and inform users how to set-up the software in the most ideal way or whether they are achieving the expected performance.
To limit the burden on developers, the aforementioned steps should be automated within continuous integration (CI) practices, introducing continuous benchmarking (CB) to it. For HPC, an added complexity is the requirement of more than the usual CI backends, with access to longer running steps and more resources than available on a single node. Reusing test cases that are easily run by hand is another simplification for developers that may not be familiar with the research field. We show our solution to CB that we use at the Juelich Supercomputing Centre (JSC), where we combine the already implemented benchmarking via the Juelich Benchmarking Environment (JUBE) with properly authenticated CI steps running on the supercomputing systems at JSC. The combined results, including the evolution over time, are then further processed and displayed on pages published via CI.

Slot length:

Poster Session / 71

Enhance agent-based model AgriPoliS with individual decision model FarmDyn through deep learning surrogate model FarmLin

Authors: Franziska Appel[1]; Hugo Storm[None]

Co-authors: Changxing Dong [2]; David Schäfer ; Linmei Shang

[1] *Leibniz Institute of Agricultural Development in Transition Economies*

[2] *Leibniz Institute for Agricultural Development in Transition Economies (IAMO)*

Corresponding Author: appel@iamo.de

AgriPoliS, developed at the Leibniz Institute of Agricultural Development in Transition Economies (IAMO) over more than two decades, is an agent-based model for simulating the development of agricultural regions under changing economical, ecological and societal conditions. Individual farms, modeled as agents in AgriPoliS, make their decisions independently but interact with each other through different markets, especially land market. FarmDyn , developed at the University of Bonn since 2013, is an extensively detailed individual farm decision model designed to simulate farms' responses to various conditions, such as changing prices or policy instruments. In addition to economic indicators, FarmDyn encompasses a diverse set of environmental metrics.
Both models have been successfully applied in various research projects in different countries and exhibit distinct strengths and limitations. FarmDyn, focusing on individual farms, offers a higher level of detail in decision-making compared to AgriPoliS. Conversely, AgriPoliS, by considering inter-farm interactions, enables the exploration of dynamic and emergent phenomena.
The objective of this project entails integrating these two models to leverage their respective strengths and mitigate their weaknesses, thereby enhancing the overall modeling framework. Essentially, our aim is to integrate FarmDyn within AgriPoliS and thus link the detailed decisions at farm level with the interactions between the farms.
Given the autonomous evolution of both models across diverse institutions and locations, their independent development has led to conceptual and technical disparities, presenting unique methodological challenges in achieving alignment. Directly embedding FarmDyn into AgriPoliS encounters several complexities. For instance, the models operate in different programming languages—FarmDyn employs GAMS (Java), while AgriPoliS is built in C++. Further, FarmDyn's reliance on GAMS necessitates a license for the mixed linear programming solver CPLEX, potentially limiting simulation environments upon direct integration. Moreover, the intricate model complexity of FarmDyn poses critical considerations regarding computational efficiency of the integrated model.
To bridge these differences, our approach first involves the conceptual alignment of the models and the standardization of input and output formats. To do this, we define an interface with detailed descriptions of input and output variables that include names, definitions, data types and permitted data ranges. This standardized interface facilitates FarmDyn's customized contributions to the AgriPoliS environment. To technically facilitate the alignment, our approach involves the utilization of a surrogate model named FarmLin —a deep neural network-based surrogate implemented in Python using TensorFlow. FarmDyn generates extensive datasets used to train FarmLin, which, upon integration, bolsters the alignment between FarmDyn and AgriPoliS, overcoming inherent disparities and enhancing the overall modeling framework.
The integrated model was tested for the "Rheinisches Revier" agricultural region in Germany. The methodology has proven to be successful as it allows a comprehensive analysis of the structural development of the region, considering the interactions between farms. At the same time, it enabled the dynamic assessment of environmental indicators that were modeled exclusively in FarmDyn. This holistic approach enables a nuanced understanding of the region's development, encompassing both inter-farm dynamics and environmental changes.

Slot length:

72

Accelerating massive data processing in Python with Heat

Author: Claudia Comito[1]

Co-authors: Fabian Hoppe [2]; Juan Pedro Gutirrez Hermosillo Muriedas [3]; Kai Krajsek [4]; Michael Tarnawa [5]

[1] *Forschungszentrum Jülich, Jülich Supercomputing Centre*

[2] *DLR - Institut für Softwaretechnologie - HPC*

[3] *SCC*

[4] *Forschungszentrum Jülich GmbH*

[5] *Forschungszentrum Jülich*

Corresponding Authors: fabian.hoppe@dlr.de, c.comito@fz-juelich.de

Manipulating and processing massive data sets is challenging. For the vast majority of research communities, the standard approach involves setting up Python pipelines to break up and analyze data in smaller chunks, an inefficient and prone-to-errors process. The problem is exacerbated on GPUs, because of the smaller available memory.

Popular solutions to distribute NumPy/SciPy computations are based on task parallelism, introducing significant runtime overhead, complicating implementation, and often limiting GPU support to specific vendors.

In this tutorial, we will show you an alternative based on data parallelism. The open-source library Heat 1 builds on PyTorch and mpi4py to simplify porting of NumPy/SciPy-based code to GPU (CUDA, ROCm, including multi-GPU, multi-node clusters). Under the hood, Heat distributes massive memory-intensive operations and algorithms via MPI communication, achieving significant speed-ups compared to task-distributed frameworks. On the surface however, Heat implements a NumPy-like API, is largely interoperable with the Python array ecosystem, and can be employed seamlessly as a backend to accelerate existing single-CPU pipelines, as well as develop new HPC applications from scratch.

You will get an overview of:
- Heat's basics: getting started with distributed I/O, data decomposition scheme, array operations
- Existing functionalities: multi-node linear algebra, statistics, signal processing, machine learning…
- DIY how-to: using existing Heat infrastructure to build your own multi-node, multi-GPU research software.

We'll also touch upon Heat's implementation roadmap, and possible paths to collaboration.

1 M. Götz et al., "HeAT –a Distributed and GPU-accelerated Tensor Framework for Data Analytics," 2020 IEEE International Conference on Big Data (Big Data), Atlanta, GA, USA, 2020, pp. 276-287, doi: 10.1109/BigData50022.2020.9378050.

Slot length:

other(help with comment)

Reproducible Packaging / **73**

Reproducible Package Environments for Modeling Workflows

Author: Pascal Sauer[1]

[1] *Potsdam Institute for Climate Impact Research*

Corresponding Author: pascal.sauer@pik-potsdam.de

By default languages like R and Python load packages/modules from a system-wide package environment. Thus all projects use the same package versions which makes it hard to track which package versions were used for a specific project, and using different package versions for separate projects

becomes impossible. Package updates which are required for one project may break other projects. These issues are especially problematic on large multi-user systems like an HPC (high performance computer cluster).
To solve this virtual package environments can be used. These isolate projects from each other and from the system-wide package environment, making it possible to use different package versions in separate projects. Also, this makes it clear which package versions were used for a project.
In this talk we present our package environment solution for developing scientific models on a large multi-user HPC. We describe how and why our approach is different from typical package environment setups, what the advantages and downsides are, and what lessons we have learned. In summary, at the cost of some additional complexity for users, our approach greatly improves reproducibility, robustness, and control over which package versions are used for each model run. Using a shared package cache we need 8.3 GB disk space for 3400 different package versions, and restoring entire package environments with over 200 packages takes a couple of seconds.

Slot length:

74

Automating your FAIR software publications with HERMES –a hands-on workshop

Authors: David Pape[1]; Guido Juckeland[2]; Michael Meinel[3]; Oliver Bertuch[4]; Oliver Knodel[2]; Sophie Kernchen[None]; Stephan Druskat[5]

[1] *Helmholtz-Zentrum Dresden-Rossendorf (HZDR)*

[2] *Helmholtz-Zentrum Dresden-Rossendorf*

[3] *Deutsches Zentrum für Luft- und Raumfahrt e.V.*

[4] *Forschungszentrum Jülich*

[5] *German Aerospace Center (DLR)*

Corresponding Author: sophie.kernchen@dlr.de

RSEs are required to publish reproducible software to satisfy the FAIR for Research Software Principles. To save RSEs the arduous labor of manual publication of each version, they can use the tools developed in the HERMES project. HERMES (HElmholtz Rich MEtadata Software Publication) is an open source project funded by the Helmholtz Metadata Collaboration. The HERMES tools help users automate the publication of their software projects and versions together with rich metadata. They can automatically harvest and process quality metadata, and submit them to tool-based curation, approval and reporting processes. Software versions can be deposited on publication repositories that provide PIDs (e.g. DOIs).

In this hands-on workshop, we briefly present and demonstrate HERMES before guiding RSE participants through setting up the HERMES publication workflow for their own software projects. We also cater for participants who want to deploy HERMES for their own infrastructure.

The workflow follows a push-based model and runs in continuous integration (CI) infrastructures such as GitHub Actions or GitLab CI. This gives users more control over the publication workflow compared to pull-based workflows (e.g. the Zenodo-GitHub integration). It also makes them less dependent on third-party services. Rich descriptive metadata is the key element to useful software publications. The workflow harvests existing metadata from source code repos and connected platforms. Structured metadata could for example come from a Citation File Format file or a CodeMeta file. Unstructured data could be everywhere, especially in the code or the README file. HERMES processes, collates and optionally presents the gathered data for curation to keep a human in the loop. In curation, output can be controlled and errors reduced. After approval, HERMES prepares the metadata and software artifacts for automatic submission to FAIR publication repositories.

In the course of the workshop, RSEs are enabled to employ HERMES for their own projects through following a live coding session on an example project. We will address any problems that arise

along the way and help participants solve them. Finally, we will discuss potential improvements of the HERMES workflow based on the hands-on experience participants made.

The workshop should last about 90 min. The target audience is everyone who deals with research software. Researchers, developers, curators and supervisors are welcome as well as everyone interested. No specific expertise or previous experience is needed. We work with GitHub or GitLab, and use their continuous integration tools, so some previous experience with these platforms may be helpful.

Slot length:

other(help with comment)

Continuous Integration - Intermediate / **75**

Pipeline dependencies and layered test suites

Author: Jan Philipp Thiele[1]

[1] *Weierstrass Institute Berlin*

Corresponding Author: thiele@wias-berlin.de

Supporting multiple operating systems, compilers, etc. can lead
to a combinatorial explosion with the need for a large test suite.
Similarly, a large library can have many features to test such that a full run
for one of those combinations can take a long time.

In this skill-up we will talk about a common approach to address this problem,
layering test suites and pipelines.
Then, the most important parameter combinations
are tested on every pull request while a fuller set is run regularly on a scheduled basis.

This also includes a discussion about combinatorial explosion and parameter
matrices to help decide which combinations are the most important.

Slot length:

other(help with comment)

AI/ML Research Software / **76**

Breaking down complex technology: Artificial Intelligence on Extreme Edge Networks

Authors: Manuel Schrauth[1]; Moritz Thome[1]; Torsten Ohlenforst[1]; Felix Kreyß[1]

[1] *Fraunhofer IIS*

Corresponding Author: manuel.schrauth@iis.fraunhofer.de

As scientists, we frequently encounter the challenge of conveying complex research topics to a general audience. A prime example of this complexity is the field of artificial intelligence, which is currently undergoing unprecedented advancements - and many of these evolutions are hard to grasp for those not immersed in the community. When combined with other state-of-the-art developments,

such as in the field of extreme edge computing, how can research still be made accessible to a wider public?

Our project steps into this space. We developed a novel approach that allows to distribute AI algorithms in a self-regulated manner across multiple devices within a wireless ad-hoc network. The network remains stable against connectivity fluctations as, owed to its intelligent architecture, tasks can be redistributed automatically. In this process, not only one device takes over the execution of the neural network, but multiple nodes compute the task cooperatively. In order to demonstrate this technology, we built an interactive hardware demonstrator. It allows attendees to experience first-hand how their handwriting samples are processed, analyzed, and evaluated through a distributed computing process, which moreover can be actively controlled by regulating the individual devices during the action.

In our talk, we elaborate on the building process of this demonstrator. We reflect our experiences and challenges that emerged during the development, from the early concept stage up until the final continuous code integration phase. We will share insights into the practical hurdles we encountered, the software architecture and programming languages we used, the hardware solutions we adopted, and the lessons we learned along the way.

Slot length:

Continuous Integration - Intermediate / **77**

Maintainable up-to-date Documentation with CI

Author: Michele Mesiti[1]

[1] *Karlsruhe Institute of Technology (KIT)*

Corresponding Author: michele.mesiti@kit.edu

Maintaining documentation up to date can be difficult.
Synchronization between the behaviour of the software and the documentation is, on the other hand, extremely important. Lack of it means that users might lose trust in the maintainers (this is even more important for the documentation of a HPC system), and its presence is crucial, for example, for Tutorial-Driven-Development, where the documentation acts as an integration test suite for the code.

In this talk I will present a view of the challenges associated with the problem, look at solutions that the community has adopted, and present the solution developed for our specific use case and discuss how CI can support this practice.

Slot length:

Poster Session / **78**

Interactive Demonstrator: Artificial Intelligence on Extreme Edge Networks

Authors: Moritz Thome[1]; Manuel Schrauth[None]; Felix Kreyß[1]; Torsten Ohlenforst[1]

[1] *Fraunhofer IIS*

Corresponding Author: moritz.thome@iis.fraunhofer.de

As scientists, we frequently encounter the challenge of conveying complex research topics to a general audience. A prime example of this complexity is the field of artificial intelligence, which is currently undergoing unprecedented advancements - and many of these evolutions are hard to grasp for those not immersed in the community. When combined with other state-of-the-art developments, such as in the field of extreme edge computing, how can research still be made accessible to a wider public?

Our project steps into this space. We developed a novel approach that allows to distribute AI algorithms in a self-regulated manner across multiple devices within a wireless ad-hoc network. The network remains stable against connectivity fluctations as, owed to its intelligent architecture, tasks can be redistributed automatically. In this process, not only one device takes over the execution of the neural network, but multiple nodes compute the task cooperatively.

Convince yourself of the functionality of our approach with our interactive demonstrator. Observe how your own handwriting is processed, analyzed and evaluated in a distributed computing process. Intervene in the action yourself by controlling and regulating the individual devices.

Slot length:

Continuous Integration - Advanced / **79**

Continuous Integration in Complex Research Software - Handling Complexity

Author: Tobias Huste[1]

Co-authors: Christian Hueser [2]; Norman Ziegner [3]; René Widera [4]; Simeon Ehrig [4]

[1] *Helmholtz-Zentrum Dresden-Rossendorf*

[2] *Helmholtz-Zentrum Dresden-Rossendorf (HZDR)*

[3] *UFZ*

[4] *Helmholtz-Zentrum Dresden - Rossendorf*

Corresponding Author: t.huste@hzdr.de

Continuous Integration (CI) is an indispensable part of modern software development. All major providers of software development platforms now offer an integrated range of resources for Continuous Integration, some of which are free to use. This offer is not sufficient for the requirements and use cases of some scientific software projects. There are scientific software projects which would like to take advantage of a large software development platform such as GitHub for the development of an open source software project and at the same time have access to the CI resources provided locally at an institution. This use case was implemented as part of HIFIS at HZDR and combines the GitHub platform with the locally available GitLab CI resources.

The presentation shows how this integration was implemented, which hurdles could be overcome, and at the same time addresses which difficulties exist. To this end, the talk goes into a practical use case, the alpaka C++ library and their specific CI implementation. The alpaka library (https://github.com/alpaka-group/alpaka) is a C++ abstraction library for accelerator development. It allows to write code once and run it on different accelerators/processor types like CPUs, GPUs and FPGAs. Therefore, it supports a wide range of processor manufacturers, through various compilers and SDKs. This results in a large set of supported software combinations that cannot be executed on the publicly available CI resources. With smart measures, the team makes use of the CI resources that have been made available and implements measures to handle the complexity and available resources.

Slot length:

RSE Research / 80

First Results on the Security Culture Survey

Author: Michael Meinel[1]

[1] *Deutsches Zentrum für Luft- und Raumfahrt e.V.*

Corresponding Author: michael.meinel@dlr.de

Together with the University of Alabama, we are replicating "An empirical study of security culture in open source software communities" (https://doi.org/10.1145/3341161.3343520) with a focus on research software engineers in Germany and the US.
The original study collected quantitative evidence on how important different aspects of security culture is in the field of open source software development.

While we acknowledge that open source software also plays an important role throughout the research community, we are also aware that there are important differences.
E.g., while participating in an open source project is largely guided by volunteering and idealism, development of research software has more personal motivations.
Also, in research there is a different framework when it comes to (grant) policies and legal frameworks.
Hence, we thought it was time to re-do this well designed study with some own extensions in the scientific community.

The survey will be open until end of 2023 (and maybe a little longer). In this talk, we want to present some preliminary results and maybe some early findings especially with the focus on the German part of the study.

Slot length:

NFDIxCS Hackathon / 81

NFDIxCS-HackaThon: Constructing a Research Data Management Container and its Platform

Authors: Marting Armbruster[None]; Jan Bernoth[1]; Michael Goedicke[None]; Michael Striewe[None]

[1] *Universität Potsdam*

Corresponding Authors: jan.bernoth@uni-potsdam.de, michael.goedicke@uni-due.de

The Research Data Management Container (RDMC) is a key element of the NFIDxCS architecture. To integrate diverse communities into the requirements engineering process, we conduct a series of HackaThons. This HackaThon is a part of the initiative, aiming to engage the community of researchers and research software engineers. Given the architecture's complexity, developing a prototype is essential to demonstrate the critical components and the overall NFIDxCS system for managing RDMCs. This practical, hands-on workshop will contribute to enhancing and expanding the conceptual version of the RDMC and its associated platform. In the workshop, we use a modularized approach to allow flexible engagement in the hacking over the course of the day and includes the following thematic areas:

1. Development environment (django, python, git)
2. Outline of the existing work: prototype in terms of concepts and implementation
3. Outline of the strategic process and the particular aims of the day
4. Brainstorming and division of tasks and efforts

5. Work with a few regular stand up meetings
6. Collection and Integration of work results
7. Further proceedings and conclusion

The state of the work on the RDMC and platform can be characterized as a prototype based on already existing work done by the authors by the end of November 2023. There will be further steps in the development leading to a kind of bare bone version the RDMC and platform realizing basic functionality realizing the concepts from (1,2).

The preparation will include a set of questions aimed at extending the basic functionality. The following list serves as a starting point to guide the discussion on specific topics chosen for the HackaThon:

- How to turn the basic version in a template mechanism including operations to instantiate a template?
- What specific mechanisms are necessary to create (sub)discipline-specific templates for predefined use cases or examples (e.g., from disciplines like Human-Computer Interaction, Software Engineering)?
- What fundamental concepts should be introduced to incorporate dimensions like privacy, access roles, and security?
- What workflows are required to ensure foolproof access for different roles in the research data lifecycle?
- Extending the existing fundamental concepts to offer additional views to foster the FAIR principles

To capitalize on the results of the HackaThon, they will be integrated into the existing prototype in a git repository.

A brief report for the general interested public will be published on the NFDIxSC (https//www.nfdixcs.org) website as well.

(1) Goedicke,Michael; Lucke,Ulrike (2022): Research Data Management in Computer Science - NFDIxCS Approach. INFORMATIK 2022. DOI: 10.18420/inf2022_112.

(2) Firas Al Laban; Jan Bernoth; Michael Goedicke; Ulrike Lucke; Michael Striewe; Philipp Wieder; Ramin Yahyapour (2023): Establishing the Research Data Management Container in NFDIxCS. Vol. 1 (2023): 1st Conference on Research Data Infrastructure (CoRDI) DOI 10.52825/cordi.v1i.395

Slot length:

other(help with comment)

82

Introduction to Test Driven Development

Author: Sven Marcus[1]

Co-authors: Jan Linxweiler [1]; Soeren Peters [1]

[1] *Technische Universität Braunschweig*

Corresponding Authors: soe.peters@tu-braunschweig.de, s.marcus@tu-braunschweig.de

Test-driven development (TDD) is an approach in software development that plays a role in enhancing the overall quality and efficiency of the software development process.

TDD involves writing automated tests before developing the actual code. In this approach, the developer begins by creating a test that intentionally fails. Subsequently, the necessary code is implemented to pass the test, followed by code refactoring to improve its overall quality. This iterative cycle is applied for each new feature or modification to the codebase.

Utilizing TDD independently or in conjunction with other practices can contribute to maintaining the code in a consistently functional and deployable state. This collaborative approach is effective in identifying and resolving issues or bugs early in the development process, ensuring a smoother and more reliable software development journey.

Slot length:

Workshop (1h)

Cross-Platform Development with C/C++ / 83

libjapi: An Open Source C Library for Seamless JSON Messaging Across Platforms

Author: Michael Baron[1]

Co-authors: Katja Vornberger [1]; Moritz Thome [1]

[1] *Fraunhofer IIS*

Corresponding Authors: katja.vornberger@iis.fraunhofer.de, michael.baron@iis.fraunhofer.de

Creating an architecture for distributed system consisting of several machines operating different software with different programming languages can be a challenge. Do you find yourself designing new interfaces over and over again with only use case specific differences? We present a solution to that: libjapi.

libjapi is an abstract and reliable C library that can be integrated in existing software to provide a configurable API. Download it from https://github.com/Fraunhofer-IIS/libjapi and use it under MIT license.

libjapi receives newline-delimited JSON messages via TCP and calls registered C functions. A JSON response is returned for each request. The benefit to you: All string handling is done by the library. Multiple users can be connected at the same time. Furthermore, it is also possible to create push services, which asynchronously push JSON messages to the clients subscribed to them. The behaviour is highly customizable at the server side. The clients can be implemented in any other language or framework as long as they can receive and transmit JSON messages over TCP. Different clients can interact with the same server as in the following use case.

We use libjapi to control the settings of an SDR (software defined radio) modem. libjapi is used by the GnuRadio based system to provide an API, that can change settings like the center frequency of the received or transmitted signals. Also measured values like received channel power and an overview of the most important settings are provided as push service. Three different clients were implemented, one command line tool in Python and a Vue based interactive web frontend. Both can operate at the same time. Additionally another service in C listens to the push service and records all these values into a database.

The library is provided as free open source software to ease the life of other developers like you. Feel free to use it, provide feedback through GitHub and add your own contributions and improvements. Ideas for future improvements include authentication, the integration of a JSON schema validator and usability improvements.

Slot length:

Lessons learned and applied / **84**

ESM-Tools - A modular approach of an Earth-System-Model infrastructure software

Author: Nadine Wieters[1]

Co-authors: Sebastian Wahl [2]; Miguel Andrés-Martínez [1]; Paul Gierz [1]; Jan Streffing [1]

[1] *Alfred-Wegener-Institut Helmholtz Zentrum für Polar- und Meeresforschung*

[2] *GEOMAR Helmholtz-Zentrum für Ozeanforschung Kiel*

Corresponding Author: nadine.wieters@awi.de

ESM-Tools is a modular infrastructure software that enables the seamless building, configuration and execution of Earth System Models (ESM) on various High Performance Computing (HPC) platforms. The software is developed at the Alfred Wegener Institute for Polar and Marine Research in Bremerhaven, jointly with the GEOMAR Helmholtz-Zentrum für Ozeanforschung in Kiel. The software is open-source and distributed through GitHub.

The aim of ESM-Tools is to provide an infrastructure tool that includes different ESM components and facilitates the use of these ESMs on different HPC systems. The software must therefore be able to handle many different possible contingencies that these models and HPC systems require. Another demand of the software is to be easily expandable in order to include future ESMs and HPCs and thus also increase the modularity of the ESMs. One of the main requirements is that all of these adaptations to the extensibility and to the functionality of the software should be customizable by the user/researcher of the software and not necessarily by an experienced software engineer. To fulfil these requirements, it must be possible to expand the functionality without changing the source code.

In order to address the above stated software requirements we applied the following design choices: (i) use of a modular software architecture, (ii) following the separation-of-concerns principle: separate source-code (consists of an HPC- and model-agnostic Python back-end) and configuration, (iii) a modular and hierarchical configuration: modular easy-to-read/write YAML files defining the configuration of each specific component of the setup (HPC- and model configuration), (iv) enable an extended functionality to the configuration files by applying a special configuration file syntax (esm-parser), (v) provide an adaptable workflow and plugin manager that is configurable by the advanced user to extend and add new functionality.

In this contribution we will introduce ESM-Tools and the design choices behind its architecture. Additionally, we will discuss the advantages of such a modular system, and address the challenges associated with its usability and maintainability resulting from these design choices and our mitigation strategies.

Slot length:

AI/ML Research Software / **85**

Towards sustainability of the legacy research CFD code VirtualFluids

Author: Soeren Peters[1]

Co-author: Jan Linxweiler [1]

[1] *Technische Universität Braunschweig*

Corresponding Author: soe.peters@tu-braunschweig.de

In many disciplines, research software is nowadays essential for scientific progress. Most often, this software is written by the scientists themselves. However, they usually pursue a short-term strategy during development, aiming at the earliest possible results. However, most often this approach leads to low software quality, especially since the scientists are generally self-taught programmers. As a result, widespread and long-term use of the software is prevented and, at the same time, the quality of scientific research and the pace of progress are compromised.

The SURESOFT project at the TU Braunschweig aims to establish a general methodology and infrastructure based on Continuous Integration (CI) for research software projects. CI is a prerequisite for improving the quality of research software, simplifying software delivery, and ensuring long-term sustainability and availability.

In this talk, I will present how we applied the ideas and concepts of SURESOFT to our research code VirtualFluids. The code is a Computational Fluid Dynamics solver based on the Lattice Boltzmann Method for turbulent, thermal, multiphase and multicomponent flow problems as well as for multi-field problems such as Fluid-Structure-Interaction including distributed pre- and postprocessing capabilities for simulations. VirtualFluids is designed to be used on High-Performance-Computing platforms with both GPGPUs and CPUs. Efficiency has always been critical, probably even more important than obtaining maintainability. As a result, in the past VirtualFluids lacked a delivery strategy as well as quality assurance. In my presentation, I'll talk about how we used ideas from SURESOFT to improve VirtualFluids and how we refactored the application to find a better balance between efficiency and making sure the software is delivered well and is high quality. This way, VirtualFluids can handle powerful computation while becoming more structured and adaptable.

Slot length:

Continuous Integration - Advanced / 86

Github and Gitlab - Combine the best of both worlds

Authors: Jakob Fritz[1]; Thomas Gruber[2]

[1] *Forschungszentrum Jülich GmbH*

[2] *Friedrich-Alexander-Universität Erlangen-Nürnberg*

Corresponding Authors: thomas.gruber@fau.de, j.fritz@fz-juelich.de

The versioning of code is important to keep track of how code changed over time. Git is the code versioning that is mainly used, and the two most popular platforms for git are Github and Gitlab.

This talk aims to show ways to combine the best of these two platforms:
The community and visibility of Github with the option for self hosting and additional Continuous Integration features offered by Gitlab.

HPC systems are commonly and CI-systems sometimes reachable only from within the network on site. In these cases the code must reside locally (e.g. on self hosted Gitlab instances) if testing shall be done on the systems. Without a synchronization the code owners need to decide on whether to do HPC-backed CI (via Gitlab) or to include their peers (via Github). The synchronization enables to do both. It therefore leads to a higher interaction with the peer group as well as being able to test the software on the systems/machines where it shall be deployed to.

This is an advanced topic for the CI track.

Slot length:

other(help with comment)

Continuous Integration - Current Research and Beyond! / 87

CI topics in research software and beyond

Authors: Florian Goth[1]; Jakob Fritz[2]; Jan Philipp Thiele[3]; Michele Mesiti[4]; René Caspart[5]; Thomas Gruber[6]

[1] *Universität Würzburg*

[2] *FJZ, JSC*

[3] *Weierstrass Institute Berlin*

[4] *Karlsruhe Institute of Technology (KIT)*

[5] *Karlsruhe Institute of Technology (KIT)*

[6] *FAU*

Corresponding Author: rene.caspart@kit.edu

Continuous Integration (CI) is an invaluable asset and an important aspect of any (research) software development and research software lifecycle.
However, CI is often only considered as a side aspect when giving talks about software projects. To remedy this, we propose a session targeting only the topic of CI and including the possibility for shorter talks, to lower the entry barrier. In addition, as CI can also be used for many other possible use-cases and scenarios beyond its application in (research) software development, we also want to offer space for these topics as part of the session.

As the format for this session, we intend to offer the possibility for short lightning talks from a wide range of participants on topics revolving around CI. These talks can cover notable and noteworthy usage scenarios and examples for CI and also encourage ideas and experiences for scenarios using CI outside of software development, e.g. for papers, websites, teaching and similar.

Among the organizers, we have already identified possible topics for this session. However, before defining the final agenda, we intend to open the session to anyone in the RSE community interested in contributing and presenting their favorite topic. To this end, we will circulate a pad and an open call for lightning talks via, e.g., the de-rse mailing list to collect interested parties and topics based on which we will compose the agenda for the session.

Slot length:

Workshop (1h)

Reproducible Packaging / 88

Improving reproducibility of scientific software using Nix/NixOS: A case study on preCICE adapters and solvers

Authors: Max Hausch[None]; Simon Hauser[None]

Corresponding Authors: st175425@stud.uni-stuttgart.de, st148883@stud.uni-stuttgart.de

Ensuring the reproducibility of scientific software is crucial for the advancement of research and the validation of scientific findings.
However, achieving reproducibility in software-intensive scientific projects is often challenging due to dependencies, system configurations and software environments.
In this paper, we present a possible solution for these challenges by utilizing Nix and NixOS.
Nix is a package manager and functional language that allows to mitigate these problems by guaranteeing that a package and all its dependencies can be built reproducibly as long as there is a build plan at the desired time.
NixOS is a purely functional Linux distribution, built on top of Nix that enables the build of reproducible systems including configuration files, packages and their dependencies.

We present a case study on improving the reproducibility of preCICE, an open-source coupling library, and some of its main adapters using Nix and NixOS.
Using this approach, we demonstrate how to create a reproducible and self-contained environment for preCICE and highlight the benefits of using Nix and NixOS for managing software and system configurations, resulting in improved reproducibility.
In addition, we compare the usability and reproducibility provided by Nix, in the context of preCICE, with two already established high-performance computing (HPC) solutions, Spack and EasyBuild.
This evaluation enables us to assess the advantages and disadvantages of employing Nix to improve reproducibility in scientific software development within an HPC context.

Slot length:

Sustainable Organisation / 89

Creating Generic Software Solutions for Specific Research Issues - Research Software Product Development without Reinventing the Wheel

Author: Zeki Mustafa Doğan[1]

Co-authors: Ingo Pfennigstorf [1]; Kristine Schima-Voigt [1]

[1] *SUB Göttingen*

Corresponding Authors: pfennigstorf@sub.uni-goettingen.de, dogan@sub.uni-goettingen.de, schima-voigt@sub.uni-goettingen.de

Certain research questions necessitate highly specialized software solutions tailored to the unique intricacies of the problem at hand. These questions often arise from the complex and nuanced nature of the research domain, demanding precise methodologies and algorithms that cater to specific requirements. In such cases, attempting to implement a generic solution might prove counterproductive and time-consuming. Nevertheless, while certain research questions demand highly specific software solutions, it is equally important to recognize instances where commonalities and synergies exist across diverse projects. In scenarios where the fundamental requirements overlap, the development of generic solutions becomes not only feasible but also advantageous. By leveraging shared expertise and identifying these common threads, research software engineers can create versatile research software products that cater to multiple research inquiries whereas reinventing the wheel leads to inefficient allocation of resources, as valuable time and effort are expended on solving problems that may have already been addressed by existing solutions. Striking the right balance between tailored and generic solutions not only optimizes resource utilization but also establishes a foundation for standardized practices, promoting collaboration and interoperability. This approach allows the research community to benefit from both the precision of specialized tools and the efficiency gained through the development of reusable, standardized software components. This nuanced approach acknowledges the diversity of research questions while harnessing the potential for synergies and collaborative advancements in research software engineering.

Creating research software products requires a combination of technical expertise, domain knowledge, effective collaboration with researchers, and a commitment to best practices. It is also imperative to incorporate the identification of overlapping requirements across diverse projects into the work culture of the RSE team.

The SUB software and service development team has been developing solutions for technical and methodological issues in project teams using collaborative and agile methods such as Scrum for over 10 years. In this session we will describe how we try to recognize synergies at an early stage, which methods and strategies we use to develop generic solutions in project work, what challenges we encounter, and how we manage them. Reinventing the Wheel

Slot length:

AI/ML Research Software / **90**

Documenting ML Experiments in HELIPORT

Authors: David Pape[1]; Oliver Knodel[2]; Sebastian Starke[3]

[1] *Helmholtz-Zentrum Dresden-Rossendorf (HZDR)*

[2] *Helmholtz-Zentrum Dresden-Rossendorf*

[3] *HZDR*

Corresponding Author: d.pape@hzdr.de

HELIPORT is a data management guidance system that aims at making the components and steps of the entire research experiment's life cycle findable, accessible, interoperable and reusable according to the FAIR principles. It integrates documentation, computational workflows, data sets, the final publication of the research results, and many more resources. This is achieved by gathering metadata from established tools and platforms and passing along relevant information to the next step in the experiment's life cycle. HELIPORT's high-level overview of the project allows researchers to keep all aspects of their experiment in mind.

A particularly interesting use case are machine learning projects. They are often prototypical in nature and driven by iterative development, so reproducibility and tranparency are a great concern. It is essential to keep track of the relationship between input data, choices in model parameters, the code version in use, and performance measures and generated outputs at all times. This requires a data management platform that automatically records the changes made and their effects. Existing MLOps tools (such as Weights and Biases, MLFlow) live entirely in the ML domain and start their workflow with the assumption that data is available. HELIPORT, on the other hand, takes care of the data lifecycle as well. Our envisioned platform interoperates with the domain specific tools already used by the scientists, and is able to extract relevant metadata (e.g. provenance). It can also make persistent any additional information such as papers the work was based on, documentation of software components, workflows, or failure cases. Moreover, it should be possible to publish these metadata in machine-readable formats.

The challenge arising from these aspects consists in integrating ML workflows into HELIPORT in such a way that they work on the provided data and metadata. The goal is also to enable the comprehensible development of ML models alongside the experiment documented in HELIPORT. This allows different teams (e.g. experimentalists and AI specialists) to work together on the same project in a seamless manner, and help generate FAIRer outcomes. In the long term we hope to aide in establishing digital twins of facilities, and making their maintenance a part of the data management proces.

Slot length:

Poster Session / **91**

neuro-conda: A Python Distribution For Neuroscience

Authors: Joscha Tapani Schmiedt[1]; Stefan Fuertinger[2]

[1] *Brain Research Institute, University of Bremen*

[2] *Ernst Strüngmann Institute (ESI) for Neuroscience in Cooperation with Max Planck Society*

Corresponding Author: stefan.fuertinger@esi-frankfurt.de

Neuroscience is a multi-disciplinary field that involves scientists from diverse backgrounds such as biology, computer science, engineering, and medicine. These scientists work together to understand how the brain operates in health and disease. The areas of application in neuroscience that require

software are as diverse as the scientific backgrounds and programming skills of the scientists, ranging from experimental control and data collection to simulations, data analysis, and management. Python has established itself as the de-facto standard in modern neuroscience due to its accessibility and broad scope of applicability.

However, the software tooling supporting Python workflows has to be handled by often inexperienced end-users leveraging well-established scientific libraries shipped across dozens, sometimes hundreds of dependent packages. Setting these up in a robust and reproducible manner is crucial for the quality of the research but oftentimes not trivial to accomplish. For example, the dependencies of one package may be incompatible with another resulting in a conflict that has to be resolved manually. Python's lack of a standardized package manager spurred the emergence of several third-party solutions, such as pip, conda, and poetry, making this task even more complex.

To ease the initial burden of dependency management, we built the Python distribution neuro-conda as an accessible entry point into the existing universe of software tools for neuroscience. It provides an easy-to-install, ready-to-use computational working environment for neuroscience supporting all major desktop operating systems (Windows, macOS, and Linux). Installation from scratch can be done with a single one-liner from the command line. Adding neuro-conda to existing conda installations is also possible. Through curation of the included packages and providing explanatory package lists, neuro-conda simplifies the setup process and ensures reproducibility of the research. It is available from https://github.com/neuro-conda.

We aim to prepare bi-annual releases that bring new feature updates of included libraries to end-users while previous releases remain available. The neuro-conda version provides a unique identifier of a complete environment, making it **citable and reproducible**. Each release is tested automatically in a continuous integration pipeline to ensure support for multiple Python versions and operating systems.

In summary, the **neuro-conda distribution bundles commonly used neuroscience packages into curated conda environments**, which are rigorously tested and validated for consistency and reliability.

Slot length:

Workflowmanagement for Parallel Computing / 92

Pathway to exascale: experiences in adopting more scalable algorithms in ESPResSo

Author: Jean-Noël Grad[1]

Co-authors: Alexander Reinauer [1]; Christian Holm [1]; Rudolf Weeber [1]

[1] *Institute for Computational Physics, University of Stuttgart, Germany*

Corresponding Author: jgrad@icp.uni-stuttgart.de

Soft matter systems often exhibit physical phenomena that resolve different time- or length-scales, which can only be captured in numerical simulations by a multiscale approach that combines particle-based methods and grid-based methods. These algorithms have hardware-dependent performance characteristics and usually leverage one of the following optimizations: CPU vectorization, shared memory parallelization and offloading to the GPU.

The ESPResSo package1 combines a molecular dynamics (MD) engine with a lattice-Boltzmann (LB) solver, electrostatics solvers and Monte Carlo schemes to model reactive and charged matter from the nanoscale to the mesoscale, such as gels, energy materials, and biological structures2. The LB method is widely used to model solvents and diffusive species that interact with solid boundaries and particles. The popularity of the method can be explained by its simplicity, re-usability in different contexts, and excellent scalability on massively parallel systems. New LB schemes can be rapidly prototyped in Jupyter Notebooks using LbmPy3 and PyStencils4, which rely on a symbolic

formulation of the LB method to generate highly optimized and hardware-specific C++ and CUDA kernels, that can be re-used in waLBerla5.

Originally designed for high-throughput computing, ESPResSo has recently found new scientific applications that require resources only available at high-performance computing (HPC) facilities. Major structural changes were necessary to make efficient use of these resources:6 replacing the original LB code by waLBerla, a library tailored for HPC; rewriting the MD engine to support data layouts optimized for memory access; and redesigning the particle management code to reduce communication overhead. These changes make ESPResSo more performant, productive and portable, and easily extensible and re-usable in other domains of soft matter physics. In collaboration with our partners of the Cluster of Excellence MultiXscale, the software is now available on EasyBuild and will be part of the EESSI[7] pilot.

References:
1 Weik et al. “ESPResSo 4.0 –an extensible software package for simulating soft matter systems”. In: European Physical Journal Special Topics 227.14, 2019. doi:10.1140/epjst/e2019-800186-9
2 Weeber et al. “ESPResSo, a Versatile Open-Source Software Package for Simulating Soft Matter Systems”. In: Comprehensive Computational Chemistry. Elsevier, 2024. doi:10.1016/B978-0-12-821978-2.00103-3
3 Bauer et al. “lbmpy: Automatic code generation for efficient parallel lattice Boltzmann methods”. In: Journal of Computational Science 49, 2021. doi:10.1016/j.jocs.2020.101269
4 Bauer et al. “Code generation for massively parallel phase-field simulations”. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, 2019. doi:10.1145/3295500.3356186
5 Bauer et al. “waLBerla: A block-structured high-performance framework for multiphysics simulations”. In: Computers & Mathematics with Applications 81, 2021. doi:10.1016/j.camwa.2020.01.007
6 Grad, Weeber, “Report on the current scalability of ESPResSo and the planned work to extend it”. MultiXscale Deliverable, EuroHPC Centre of Excellence MultiXscale, 2023. doi:10.5281/zenodo.8420222
[7] Dröge et al. “EESSI: A cross-platform ready-to-use optimised scientific software stack”. In: Software: Practice and Experience 53(1), 2023. doi:10.1002/spe.3075

Slot length:

Poster Session / **93**

cff2pages: Generate a frontend from your metadata

Author: Jan Bernoth[1]

[1] *Universität Potsdam*

Corresponding Author: jan.bernoth@uni-potsdam.de

In discussions about FAIR software, primary use cases often center on making research accessible to other participants. The focus is usually on metadata. Metadata are used to make artifacts findable and accessible for search engines. This is structured in machine-readable file formats, based on YAML or JSON, and acts as intermediaries for researchers searching for artifacts. However, these formats aren’t always user-friendly, which might discourage researchers from engaging with the content, despite its potential value.

Presenting metadata in a human-readable form, like a webpage, could help all researchers get an overview of a project. While there is an existing website generator named ya2RO 1 that automates generation using a minimal YAML description, it relies on a new YAML scheme. This scheme requires users to learn and a community to maintain it. Without an active user community, there’s a high risk that the scheme becomes outdated or fails to meet user needs. An established file format like CFF 2 for feeding metadata into a webpage generator could ensure continuous updates and leverage an existing user base for initial adoption. This concept led to the creation of ‘cff2pages’. The tool generates a single webpage displaying the title, authors and their affiliations, keywords, persistent identifier, license, and an abstract from the CFF. This page can easily be integrated into

CI/CD pipelines for creating GitHub/GitLab Pages or webpages for internal repositories within institutions.

Currently, cff2pages is in a prototype stage, encouraging early adopters to provide use cases for wider integration into research systems. It offers potential enhancements, such as incorporating semantic web-oriented metadata (e.g., RO-Crate3) or integration into workflows like HERMES 4. To suit institutional publication processes, the template is customizable to match the corporate design of specific institutions.

1. A. F. Pavel and D. Garijo, "Ya2ro: A tool for creating Research Objects from minimum metadata," 2023.
2. S. Druskat, J. H. Spaaks, N. Chue Hong, R. Haines, and J. Baker, "Citation File Format (CFF) - Specifications,"2021.
3. S. Soiland-Reyes et al., "Packaging research artefacts with RO-Crate,"2021, doi: 10.48550/arXiv.2108.06503
4. Meinel, M., Druskat, S., Kelling, J., Bertuch, O., Knodel, O., & Pape, D. hermes (Version proof-of-concept) (Computer software)

Slot length:

Local RSE-related Organisations / 94

The natESM sprint concept: a tool for advancing ESM software

Authors: Wilton Jaciel Loch[1]; Jörg Benke[2]; Sergey Sukov[2]; Catrin Meyer[2]; Iris Ehlert[1]; Enrico Degregori[1]

[1] *Deutsches Klimarechenzentrum (DKRZ)*

[2] *Jülich Supercomputing Centre (JSC)*

Corresponding Author: loch@dkrz.de

In Earth System modelling (ESM), the high variety and complexity of ESM processes to be simulated as well as the specialisation of scientists led to the existence of a multitude of models, each aiming on the simulation of different aspects of the system. This scenario gave rise to a highly diverse ecosystem of ESM software, whose components are written in different languages, employ different HPC techniques and tools, and overlap or lack functionalities.

To use the national technical resources and the scientific expertise more efficiently, the natESM project aims to establish a coupled seamless ESM system by providing so-called technical-support sprints. A sprint consists of a goal-oriented package of work executed by a dedicated research software engineer on a selected ESM model during a defined amount of time.

The scientist's sprint proposals undergo a technical evaluation, which is divided in two steps: The first is a "sprint check", consisting of a low-threshold method for the presentation of a sprint idea. After an open discussion about the technical feasibility and possible adjustments, if the work aligns in general with the natESM strategy, the proposals move to the second phase, with the creation of a full sprint application. This consists of a detailed document which contains the model characteristics and formalises the goals, timeline, and criteria for sprint fulfilment.

Once a scientific approval is also granted, the work on a sprint starts and lasts from a couple of weeks up to six months, depending on the stipulated timeline and objective. The period and timeline can also be adjusted depending on impediments found along the way. The type of work done is inline with natESM goals, and sprints are usually focused on – but not limited to: architecture porting, model coupling and interfacing, modularization, as well as general software engineering improvements. The overarching concept is to efficiently enable the models to progress in the desired direction, and therefore entails identifying the minimum amount of work which will allow the model to take the largest strides towards the community goal. Upon finalisation of a sprint, a

public documentation is published, potentially serving as help to other models on addressing similar problems.

The complete process is based on an open and interactive discussion between the model scientists and the research software engineers. It usually takes the form of short regular meetings and chat-based conversations. The underlying reasoning is to minimise the work a scientist has to do to communicate their problems and to receive support.

Based on the positive results observed and feedback from the community, the sprint concept is proving to be a highly effective approach to improving the technical resources of the scientific community. This tool, guided by a comprehensive strategy outlined by scientific requirements, will help the Earth system modelling future goals to be ever closer. Its applicability extends beyond the field of Earth system modelling and may also prove valuable to other research areas seeking to establish a research-software-engineering service and accelerate the results of partnerships between scientists and RSEs.

Slot length:

Teaching RSE Skills / 95

A coherent curriculum track of RSE skills for simulation software

Author: Gerasimos Chourdakis[1]

[1] *Technical University of Munich*

Corresponding Author: gerasimos.chourdakis@tum.de

What if your graduate programme actually prepared you with all the software engineering skills you need to participate in the research software community?
The M.Sc. Computational Science and Engineering (CSE) at the Technical University of Munich gathers STEM graduates from all over the world and teaches them elements of numerics, computer science, and applications. CSE graduates are the perfect candidates for developing the next PETSc, OpenFOAM, or Tensorflow. And yet, the programming-related part of the curriculum needed some aligning and dusting.

Over the past six years, we had the opportunity to look at the big picture, redesigning several courses that "tell a story" together. Nowadays, a CSE student can follow a coherent track that prepares them for working as software engineers in an RSE team developing simulation software. We start by preparing the ground with fundamental Linux, Git, Matlab/Octave, C++, and teamwork concepts in the 1-week onboarding course "CSE Primer". In the afternoons of that week, students also work in teams, developing small projects analyzing climate data.

For the rest of the semester, CSE students have to follow "Advanced Programming", a course with an ambitious name which we put a lot of effort in justifying. With the "advanced", we aim to raise the level of the inexperienced, while still offering enough opportunities for already experienced students to grow. The material covers a pragmatic mixture of modern C++ with just enough references of legacy features to be able to work with existing codebases. The slides include code snippets that the students can interact with using the Compiler Explorer. The tutorial exercises include common tools that support the development, including a debugger, sanitizer, formatter, build tools, testing frameworks, and more. An optional project lets students develop their own idea in pairs, or contribute to existing open-source projects, while participating in a code peer-review process. The lectures and tutorial are hybrid, and the in-person exam is supported by TUMExam, a system that offers digital correction and review features. The redesign of this course attracted several students from additional study programs, with the original audience of CSE students now representing less than 10% of the exponentially-growing total audience.

After the first semester, students follow a practical (lab) course. One highlight is the Computaional Fluid Dynamics Lab, in which students work in groups to implement worksheets and their own final

project, in a bare-minimum C++ PDE framework, receiving code reviews on GitLab and maving their first steps towards parallel programming and performance optimization. Cross-references between Advanced Programming and CFD Lab make the two courses coherent, without discouraging external students to join. The (not offered anymore due to staff shortage) seminar Partitioned Fluid-Structure Interaction lets students expand their research skills specific to CFD, writing their own paper and participating in peer-reviews.

This talk will give an overview of these courses, discussing several didactical and technical elements applied in each, concluding with not-so-obvious good practices.

Slot length:

Parallelization and HPC Infrastructure / **96**

lattice QCD software development for heterogeneous supercomputers

Author: Bartosz Kostrzewa[1]

Co-authors: Damian Alvarez [2]; Simone Bacchio [3]; Kate Clark [4]; Ahmed Fahmy [2]; Jacob Finkenrath [5]; Marco Garofalo [6]; Andreas Herten [2]; Balint Joo [7]; Ferenc Pittler [3]; Simone Romiti [6]; Aniket Sen [6]; Kay Thust [2]; Carsten Urbach [6]; Mathias Wagner [8]; Evan Weinberg [8]; Dean Howarth [9]

[1] *High Performance Computing & Analytics Lab, University of Bonn*

[2] *Juelich Supercomputing Center*

[3] *Cyprus Institute*

[4] *NIVIDIA*

[5] *University of Wuppertal*

[6] *HISKP, University of Bonn*

[7] *ORNL*

[8] *NVIDIA*

[9] *Berkeley National Laboratory*

Corresponding Author: bkostrze@uni-bonn.de

What does it take to develop and maintain a research code for the stochastic simulation of the physics of the strong interaction in Lattice Quantum Chromodynamics (LQCD)? How to make it run on the fastest supercomputers in the world? How many people are involved and what do they contribute when and how? What kind of development and interaction structures are useful? Which kinds of challenges need to be overcome?

In this contribution we present the ongoing and organically cooperative effort between LQCD groups in Bonn and Cyprus, the development team of the QUDA LQCD library and staff at Juelich Supercomputing Center (JSC) in enabling the tmLQCD software framework, the workhorse of the Extended Twisted Mass Collaboration, to successfully run on supercomputers with accelerators by NVIDIA and AMD today as well as machines by these and other vendors in the future.

LQCD has historically been a trailblazer discipline in its adoption of new supercomputing architectures. Some groups have even actively contributed to the development of new machines such as the BlueGene line of supercomputers. At some sites, LQCD practitioners use very large fractions of the total available computing time, such that efficiently implementing the underlying algorithms has a significant impact not only on the possible science output but also on the associated energy consumption.

Many software frameworks for LQCD have been developed by small collaborations of "user-developers" who were able to efficiently target the relatively slowly changing architectures of the time with comparatively simple algorithms, mostly just making use of MPI for inter-process communicaton and

some level of hardware specialisation via compiler intrinsincs or even inline assembly to target particular hardware architectures. The rapid proliferation of many-core and accelerated supercomputing systems by multiple vendors and the increased complexity of state-of-the-art algorithms have changed this irrevocably.

In order for LQCD codes to target current and future supercomputers, close interaction between hardware vendors, supercomputing centers as well as library and application developers is mandatory. In addition to addressing performance on individual architectures, questions about correctness testing, provenance tracking, performance-portability, maintainability and programmer productivity arise. Finally, the complexity of these new architectures leads to interesting failure modes which may be difficult or impossible for the developers to diagnose on their own.

We focus on collaborative aspects such as the open development practices of both QUDA and tmLQCD, the interaction between the QUDA development team and the LQCD community, the early access programme organised by JSC for the Juwels Booster supercomputer and the effort required over many months for the diagnosis of a particularly vexing issue with node failures on that machine. In this process we analyse which structural and interactional factors we believe have enabled us to successfully tackle these challenges at different stages and attempt to use our example to characterise what it takes to develop software for LQCD research today.

Slot length:

Poster Session / **97**

Saving Deprecated Infrastructure: GNU/Linux-enabled Reuse of macOS Workstations

Authors: Tomas Stary[1]; Matthias Schaufelberger[1]; Marie Houillon[1]; Axel Loewe[1]

[1] *Karlsruhe Institute of Technology*

Corresponding Author: tomas.stary@kit.edu

Introduction:

Product life span is an important aspect to consider when we design computing infrastructure. Short release and support life-cycles constitutes a business strategy of many hardware manufacturers to incentivize buying new products, while the hardware is still functional. Additionally, due to the restrictive terms of proprietary software licenses, it is often impossible to find independent support, which could render the system unusable.

Methods:

We faced this issue after the drop of support for macOS on ten desktop Apple computers used as students workstations. By analyzing the load of the hardware we found that the system has sufficient safety margin for the computing load of the machine. Since computationally expensive tasks are performed on HPCs, the computers will likely be used for prototyping and office work, for which their hardware is still sufficient.

Results:

Hence we decided to replace the original operating system by Ubuntu, a free and open source software (FOSS) GNU/Linux distribution. A dual-boot setup was chosen and Ubuntu was installed alongside macOS. The system was configured to work with network infrastructure similar to the previous setup (e.g. Active Directory login, shared network drives). Further, installing a free and open source software (FOSS) system enabled the usage of up-to-date security updates with an auditable code without vendor lock-in.

Conclusion:

In conclusion we succeeded to place the base for the extension of the lifetime of the computers by several years, saving considerable amount of money, but more importantly lowering the ecological

impact that purchasing a new computers would have. However, this was only possible because the computer firmware was not locked by the manufacturer and we could find appropriate drivers for the hardware peripherals. Those prerequisites might not be granted in other electronic devices and so it is important to consider this aspect when purchasing new hardware. At the same time, it is also necessary to provide support for users unfamiliar with GNU/Linux environments. We therefore plan to organize demonstrations and workshops for the users and evaluate its acceptance at a later stage.

Slot length:

Policies and Legal Aspects / **98**

Balancing the Access to Science and Competing Interests in Software Licenses

Authors: Tomas Stary[1]; Matthias Schaufelberger[1]; Marie Houillon[None]; Axel Loewe[1]

[1] *Karlsruhe Institute of Technology*

Corresponding Author: tomas.stary@kit.edu

Article 27 of the Universal Declaration of Human Rights establishes everyone's right to participate and benefit from the advancement of science. At the same time, it articulates the right to protection of authors' moral and material interests resulting from their scientific activity. Copyright laws grant exclusive rights over novel creative work to its authors who can then decide on the conditions of use, distribution and adaptation of their work. It is up to them to reconcile the apparent conflict between own interests and those of their communities in the terms of the license.

The principles of Open Science require the publication of the data in a widely accessible way without requiring the authors to allow others to adapt or modify the work (for example in the CC-BY-ND license). When we apply those principles to research software, it is also necessary to consider specific characteristics of software. As the function of software is to instruct our devices and machines to work for us, it can be viewed as a digital tool that acts in the way it was programmed. It is the best interest of the users to fully control the software they depend on. This can only be achieved if the license grants the user the freedom to run, copy, distribute, study, change and improve the software, which was formalized as four essential freedoms by the Free Software Foundation. Open Source Initiative later rephrased this conditions in ten criteria defining open source. In contrast to Open Science allowing anyone to modify the program without discrimination to any group or activity is provided in all licenses approved for free and open source software (FOSS).

Another characteristic of software is the functional difference between source code on one hand and obfuscated or machine code on the other. The source code is required for effective modification of the program for a software developer, while the obfuscated or machine code is only useful to execute the instructions specified in the software on a computer. To assure that the program and all its derivatives remain FOSS in any of the future versions a concept called copyleft was invented. The copyleft is a legal requirement that demands the release of the source code if the program is distributed. This way it safeguards the users against malicious actors and allows continuous collaboration of all contributors.

In this presentation we will give an overview of a few common FOSS licenses (such as MIT, Apache, LGPL, GPL, and AGPL) and analyze the main differences between terms and conditions in each of them. We will also discuss the possibilities of licensing research software that strike the right balance between everyone's essential rights and freedoms.

Slot length:

99

Enhancing Research Software Sustainability through Modular Open-Source Software Templates

Author: Philipp Sebastian Sommer[1]

Co-authors: Björn Lukas Saß [1]; Markus Benninghoff [1]

[1] *Helmholtz-Zentrum Hereon*

Corresponding Author: philipp.sommer@hzg.de

Research software development is crucial for scientific advancements, yet the sustainability and maintainability of such software pose significant challenges. In this tutorial, we present a comprehensive demonstration on leveraging software templates to establish best-practice implementations for research software, aiming to enhance its longevity and usability.

Our approach is grounded in the utilization of Cookiecutter, augmented with a fork-based modular Git strategy, and rigorously unit-tested methodologies. By harnessing the power of Cookiecutter, we streamline the creation process of research software, providing a standardized and efficient foundation. The fork-based modular Git approach enables flexibility in managing variations, facilitating collaborative development while maintaining version control and traceability.

Central to our methodology is the incorporation of unit testing, ensuring code integrity and reliability of the templates. Moreover, we employ Cruft, a tool tailored to combat the proliferation of boilerplate code, often referred to as the "boilerplate-monster." By systematically managing and removing redundant code, Cruft significantly enhances the maintainability and comprehensibility of research software. This proactive approach mitigates the accumulation of technical debt and facilitates long-term maintenance.

The open-source templates are available at https://codebase.helmholtz.cloud/hcdc/software-templates/. In the first 30 minutes of the tutorial, participants will gain insights into the structured organization of these software templates, enabling them to understand the framework's architecture and application to their own software products. The subsequent 30 minutes will be dedicated to a hands-on tutorial, allowing participants to engage directly with the templates, guiding them through the process of implementing and customizing them for their specific research software projects.

Maintaining research software presents distinct challenges compared to traditional software development. The diverse skill sets of researchers, time constraints, lack of standardized practices, and evolving requirements contribute to the complexity. Consequently, software often becomes obsolete, challenging to maintain, and prone to errors.

Through our tutorial, we address these challenges by advocating for the adoption of software templates. These templates encapsulate best practices, enforce coding standards, and promote consistent structures, significantly reducing the cognitive load on developers. By providing a well-defined starting point, researchers can focus more on advancing their scientific endeavors rather than grappling with software complexities.

Furthermore, the utilization of software templates fosters collaboration and knowledge sharing within research communities. It encourages the reuse of proven solutions, accelerates the onboarding process for new contributors, and facilitates better documentation practices. Ultimately, this approach leads to a more sustainable ecosystem for research software, fostering its evolution and ensuring its relevance over time.

In summary, our tutorial offers a practical and comprehensive guide to creating and utilizing software templates for research software development. By harnessing Cookiecutter with Git-based modularity, unit testing, and the power of Cruft, we aim to empower researchers in building robust, maintainable, and sustainable software, thereby advancing scientific progress in an efficient and impactful manner.

Slot length:

Workshop (1h)

Poster Session / 100

Applying FAIR principles in the phonetic sciences

Author: Stephen James Tobin[1]

[1] *Universität Potsdam*

Corresponding Author: stephen.tobin@uni-potsdam.de

The credibility of research findings in the phonetic sciences, as in any other branch of scientific inquiry, depends critically on the possibility to verify these findings and the methods used to obtain them. A commitment to peer review constitutes a pledge to uphold this principle. However, review procedures vary considerably among journals, both within and between disciplines, as do editorial boards'expectations of reviewers, and reviewers'expectations of the manuscripts and supplementary materials they evaluate. As such the peer-review process does not provide a uniform standard. Similarly, verification of some elements of the methods (e.g., statistical analyses and code/software) may depend on expertise beyond that of many reviewers within a particular discipline.

By applying the FAIR principles (Findable, Accessible, Interoperable and Reusable principles), researchers can ensure that their research data and methods remain available for review as scientific standards evolve and in spite of potentially changing institutional affiliations of contributing researchers. Application of these principles additionally maximizes the value and potential impact of the research contributions, since they can be more readily adopted by others.

In the present project, I assess and build on a recent publication that introduced a novel approach for the normalization and analysis of acoustic-phonetic data collected in investigations of phonetic accommodation—the adaptation of speech production patterns resulting from verbal interaction with other speakers. Effects of phonetic accommodation are often found, they are typically small in magnitude. Normalization may help ensure that these effects are reliably found when present. While code snippets for replication of the statistical analyses were provided, neither the data nor the normalization method were findable or accessible, thus limiting the reusability of the method. Scripts for the normalization process are made available in a Git repository, along with a specification of the prerequisite data/format, and the analytical data upon which the reported finding was based.

The current project is submitted as a demonstration of and as encouragement towards the application of FAIR principles in the phonetic sciences, both in research that is planned or under way, and where possible for already published findings.

Slot length:

Poster Session / 101

Licensing Research Software Made Easy: Introducing the License Checker-Tool

Author: Iqra Imran[1]

Co-authors: Dorothea Iglezakis [1]; Usman Ghani Amin [1]

[1] *Universität Stuttgart*

Corresponding Author: iqra.imran@ub.uni-stuttgart.de

Developing research software involves crucial decisions about software licenses, posing challenges for both researchers and developers. Assistance with issuing and verifying licenses is crucial to empower scientists to legally publish and reuse software. A license serves as a clear way to state what is allowed with a piece of software, acting as a powerful tool for software developers. Software licenses play a crucial role in defining conditions, permissions and restrictions associated with code usage. However, it is also a legal document, and only a few research software engineers feel confident interpreting legal texts. Despite established open-source licenses, challenges persist, especially when research code includes external libraries with predefined licenses. The interoperability of open-source research software relies on the compatibility of licenses. Developers frequently integrate code from different sources to enhance functionality. Understanding and recommending licenses that align with the project's goals is crucial to seamlessly incorporate external contributions without risking licensing conflicts.
This contribution introduces the License Checker, a web tool designed to simplify the complexities of selecting an appropriate license and finding the compatible licenses, benefiting both research software developers and its users. The License Checker is developed within the ReSUS Project (Reusable Software University of Stuttgart) 1, integrated into the OpenTOSCA environment 2 and a part of a platform that manages the publication, search, citation, and automated deployment of research software. The tool can be used also standalone and serves research software developers by aiding in the discovery of an appropriate license for their software. Simultaneously, it assists users in finding compatible licenses, allowing them to seamlessly incorporate external piece of software codes without risking licensing conflicts. Existing code can be uploaded to the tool and will be scanned for available license files with the help of fossology, an open source license compliance software system and toolkit 2. Based on a license ontology, the License Checker then takes this list of used licenses as input and provides a list of licenses as output that contains only licenses compatible with each of the input licenses.

In a talk (or alternatively a demo), we would like to present the current state of the License Checker-Tool and discuss the requirements of research software engineers around the licensing of their code.

References:
1. https://www.izus.uni-stuttgart.de/en/fokus/fdm-projekte/resus/
2. https://www.opentosca.org/
3. https://www.fossology.org/

Slot length:

Research Software for Computing and Visualising Text / 102

WiKoDa: A dashboard to enhance science communication

Author: Jan Bernoth[1]

[1] *Universität Potsdam*

Corresponding Author: jan.bernoth@uni-potsdam.de

WiKoDa: A dashboard to enhance science communication

Science communication is a crucial topic for both researchers and public discussion. On the one side, researchers need to develop communication skills to make their research accessible to an audience outside their field. On the other side, scientific opinion often consolidates arguments and helps audiences understand the current state of knowledge on specific discussions. Successful communication about research is a combination of ' appropriate skills, media, activities, and dialog' 1, which researchers need to strengthen. As a part of a dashboard supporting third mission activities 2, the aspect of choosing the right medium is supported by WiKoDa (a german acronym for **Wi**ssenschafts**Ko**mmunikations**Da**shboard, translating to 'Science Communication Dashboard').

WiKoDa is designed to simplify the complexity of the media landscape and to display key information for researchers, tailored to their interests. In the frontend 3, various data visualization panels summarize the media interest in a particular research topic over time, show which medium engages in research areas, and indicate whether the topic is more prevalent in regional or national media. The backend system 4 is separated from the frontend by design, allowing data processing while the frontend manages user interactions. Backend tasks include extracting data from articles, integrating data into a database, and providing an API with aggregated data. Currently, the project is in a proof-of-concept phase at the University of Potsdam. WiKoDa utilizes articles from a university-tailored press review. Future expansions could extend its utility to other institutions.

The evaluation required a simulation environment because the limited data integration may lead to certain scientific fields having no data for evaluation. This could introduce a negative bias, as users might not be able to assess functionality without data. To address this, an article creation algorithm was developed, simulating various atmospheres (rising, falling, or consistent media interest) for each research topic by generated articles and placing these generated articles in a medium. This Wizard-of-Oz simulation allows users to believe they are experiencing a real scenario, enabling them to evaluate the dashboard based on its visualizations and functionality. The evaluation indicated that participants find the dashboard useful, but some aspects of its functionality and transparency were criticized.

1. T. W. Burns, D. J. O'Connor, and S. M. Stocklmayer, "Science Communication: A Contemporary Definition,"Public Underst Sci, vol. 12, no. 2, pp. 183–202, 2003, doi: 10.1177/09636625030122004.
2. Jan Bernoth and Ulrike Lucke, "The Transfer Dashboard: Integrating the Third Mission into the University Infrastructure,"in European Journal of Higher Education IT, 2020.
3. Jan Bernoth, Lewin Kästner, Jessica Scharrenberg, Martin Schwenke, Finn Ziehe, Ulrike Lucke, and Lovis Trüstedt, "WiKoDa - Frontend,"2022, doi: 10.17605/OSF.IO/RQ8D4.
4. Jan Bernoth, Lewin Kästner, Martin Schwenke, Finn Ziehe, Lovis Trüstedt, Jessica Scharrenberg, and Ulrike Lucke, "WiKoDa - Backend,"2022, doi: 10.17605/OSF.IO/NCP3D.

Slot length:

RSE in Digital Humanities / 103

Software Applications for Individuals with Low Literacy - Key Insights and Takeaways in Research Software Engineering from an Interdisciplinary Project

Author: Jan Kuester[1]

[1] *University of Bremen*

Corresponding Author: s_ufzdc2@uni-bremen.de

Introduction

In 2018, the LEO study identified approximately 6.2 million individuals between the ages of 18 to 64 in Germany with very low literacy skills. These individuals are deemed to have a literacy level that restricts their participation in crucial aspects of society. 1 Various former and recent studies, including VERA 8, PISA, ULME I-III, and IQB, have highlighted a similar situation within the realm of apprenticeship and for juveniles in general. 5 A widely accepted public measure to enhance this situation involves implementing literacy courses at German community colleges, also known as Volkshochschule (VHS). Nevertheless, evaluating initial skills is a time-consuming process, and maintaining ongoing documentation of the learning requires continuous alignment with the comprehensive competency model created during the original "lea" (literacy education for adults; own translation) project. 2 Moreover, there is a need for anonymous self-learning based on domain-specific parameters for those affected.

lea.online

The “lea.online” project (2018-2022; BMBF FKZ W143600) had the objective of enhancing the material and competency model towards vocational fields and to give teachers exclusive function to monitor and evaluate diagnostic findings over time. Additionally, it should provide a non-intrusive anonymous interactive and gamified experience for learners. 3
The project started with digitizing the competency model, which evolved into a comprehensive content management system known as the “lea.Backend”. It functions as the backbone for the user-facing software applications: “otu.lea,”a diagnostics app accessible through a browser, “lea.Dashboard,” a browser-based analytics app for teachers, and the “lea.App,”an anonymous self-learning app available for mobile devices. 5
The lea.online project embraced a highly interdisciplinary approach to research software engineering. It comprises multiple target user groups: individuals with low literacy, teachers, and researchers, each segment defining their own requirements. The material for learning and assessment entails multiple occupational classifications spread across various subject dimensions, including reading, writing, language understanding, and mathematics. The team comprised of professionals hailing from diverse disciplines such as Educational Sciences, Mathematics, UX Research, Software Engineering, and Law with a shared goal of crafting software that is suitable for practical application in the field.

Key Insights and Takeaways

It is important to note that developing software for field use, rather than for exclusive expert use, introduces a wide range of additional factors and prerequisites typically associated with consumer-grade software designed for the mass market.
Therefore, the primary objective of this work is to present a comprehensive collection of key insights and takeaways acquired during the four-year developmental period, as well as in the current operational phase. It covers pitfalls, methods, and recommendations for future projects. It also displays examples and related code to explicitly demonstrate functionality. Due to its interdisciplinary nature, categorizing it by topics such as conceptual, UI/UX/accessibility, technical, legal, and ethical will support its overall structure.
At the end, a critical appraisal of the constraints of this project will be included.
Additionally, a roadmap for the current research and development will be presented, which offers prospects for joint efforts and progress for personal research and development.

Slot length:

Research Software for Computing and Visualising Text / 104

Combining file-based with RDMS-based scientific workflows using the LinkAhead-crawler

Authors: Henrik tom Wörden[1]; Florian Spreckelsen[1]; Stefan Luther[2]; Ulrich Parlitz[2]; Alexander Schlemmer[2]

[1] *IndiScale GmbH*

[2] *Max Planck Institute for Dynamics and Self-Organization, Göttingen*

Corresponding Author: alexander.schlemmer@ds.mpg.de

While research data management systems (RDMSs) provide many benefits for scientists, data integration is still one of the major bottlenecks for the adoption of an RDMS. Especially the omnipresent dependency on file-based digital workflows and the strong heterogeneity of file and data layouts pose important challenges. We have developed a crawler-based concept 1 that allows us to combine file-based digital workflows with RMDS-software in a way that they can be used simultaneously. Furthermore, the concept includes a flexible configuration of data integration procedures in a YAML-based format that facilitates its application to different use cases. We demonstrate how to apply these concepts practically using the LinkAhead-crawler framework (CaosDB was recently

renamed to LinkAhead). The software is published as Open Source software under AGPLv3 and can be accessed online (https://gitlab.com/linkahead/linkahead-crawler).

1 Tom Wörden, H.; Spreckelsen, F.; Luther, S.; Parlitz, U.; Schlemmer, A. Mapping hierarchical file structures to semantic data models for efficient data integration into research data management systems. Preprints 2023, 2023081170. https://doi.org/10.20944/preprints202308.1170.v1

Slot length:

RSE Practices / **105**

Behind the Scenes at Chemotion: A Popular Research Software Project

Author: Shashank Shekhar Harivyasi[1]

[1] *Karlsruhe Institute of Technology*

Corresponding Author: shashank.harivyasi@kit.edu

Chemotion ELN is a widely used Electronic Lab Notebook that promotes FAIR research. It does so by providing a comprehensive platform that helps a researcher at all stages - from planning to publishing, while automatically managing their research data for them. Funded by NFDI4Chem, the ELN's use has been growing steadily - an aspect of the project that has brought its own challenges.

In this behind the scenes talk, I will discuss our learning curve as we grew from being an in-house solution to a national one and now head towards providing the ELN as a Service. Some aspects that I will cover include moving away from monolithic architecture, solving on-site deployment challenges, experience with hiring external developers, the cost of providing 'free' support, striving for funding and our ongoing attempts at improving overall SysAdmin and end-user experience. The aim of the talk is to highlight problems in RS development (with focus on Germany), even when enough funding and well-intentioned support is available.

Slot length:

Cross-Platform Development with C/C++ / **106**

Large-scale C/C++ code restructuring with Coccinelle

Author: Michele Martone[1]

Co-author: Julia Lawall [2]

[1] *Leibniz Supercomputing Centre*

[2] *Inria Paris, France*

Corresponding Author: michele.martone@lrz.de

Did you inherit a huge C or C++ research code?
Are you supposed to make it faster (say you're in HPC)?
Are you supposed to introduce new features (say you're in physics)?
Or perhaps you want to rejuvenate this code?
You estimate the code to be too large for that to be done properly or cleanly?
This talk introduces the ideas of the Coccinelle system for large-scale code analysis and restructuring.
You are invited to visit our 2h tutorial to get a working introduction to the tool usage.
See https://github.com/coccinelle/coccinelle for more about Coccinelle.

Slot length:

other(help with comment)

107

Intro in Large-scale C/C++ code restructuring with Coccinelle

Author: Michele Martone[1]

Co-author: Julia Lawall [2]

[1] *Leibniz Supercomputing Centre*

[2] *Inria Paris, France*

Corresponding Author: julia.lawall@inria.fr

NOTE: 15:20-17:10
Did you inherit a huge C or C++ research code?
Are you supposed to make it faster (say you're in HPC)?
Are you supposed to introduce new features (say you're in physics)?
Or perhaps you want to rejuvenate this code?
You estimate the code to be too large for that to be done properly or cleanly?

This hands-on tutorial introduces concepts of the Coccinelle system for large-scale code analysis and restructuring.
In these two hours, we will teach you what you need to start working with Coccinelle.
What you need is a Linux installation or VM with a working version of Coccinelle.
Basic Unix usage and C/C++ knowledge required.
See https://github.com/coccinelle/coccinelle

Slot length:

other(help with comment)

108

Teaching RSE Working Meeting

Authors: Florian Goth[1]; Gerasimos Chourdakis[2]; Heidi Seibold[3]; Jan Linxweiler[4]; Jan Philipp Thiele[5]; Jean-Noël Grad[6]; Jeremy Cohen[7]; Leyla Jael Castro[8]; Magnus Karl Moritz Hagdorn[9]

[1] *Universität Würzburg*

[2] *Technical University of Munich*

[3] *IGDORE*

[4] *Technische Universität Braunschweig*

[5] *Weierstrass Institute Berlin*

[6] *University of Stuttgart*

[7] *Imperial College London*

[8] *ZB MED Information Centre for Life Sciences*

[9] *Charité Universitätsmedizin Berlin*

Corresponding Author: florian.goth@physik.uni-wuerzburg.de

In this workshop, the TeachingRSE project will work on how to institutionalize the education of RSEs in Germany. To that end we plan to showcase the current status to workshop participants to collect feedback on the current work, but also enable deRSE24 participants to contribute to the next publications and become regular contributors to the project.

Slot length:

Workshop (3h)

Local RSE-related Organisations / **110**

WestAI Service Center

Author: John Arnold[1]

[1] *RWTH Aachen University*

Corresponding Author: john.arnold@rwth-aachen.de

Artificial Intelligence (AI) transforms various industries with its potential for intelligent automation, leading to the creation of innovative products and services. This technological paradigm shift poses unique challenges and opportunities for research software engineers, pushing them to integrate AI into their software to maintain academic competitiveness.

In response to these evolving needs, the WestAI service center, funded by the BMBF, was established in Germany. WestAI represents a consortium of institutions, each recognized for their expertise in machine learning, artificial intelligence, and scalable computing. The center offers support to businesses and research groups aiming to integrate AI into their products and workflows. To achieve this, WestAI adopts three key strategies:

1. **Research and Development:** WestAI researchers are committed to developing open, extensive, and multimodal datasets and models suitable for a wide range of applications. Their work focuses on deepening the understanding of AI models and culminates in the creation of models, datasets, and knowledge that will be made publicly available.
2. **Consultation and Training:** The service department at WestAI aids research groups and industry partners in the adoption of AI. Services range from initial project feasibility consultations to implementation guidance and prototype development. Additionally, WestAI provides training, courses, and best practices in relevant areas.
3. **Compute Resources:** To support these initiatives, WestAI offers robust, scalable hardware resources and corresponding AI frameworks.

This presentation will offer a comprehensive overview of WestAI's services and infrastructure, demonstrating how they can advance AI projects. For further information and to engage with our experts, please visit westai.de.

Slot length:

RSE in Digital Humanities / **112**

Accessible Multimodal Editions: Rewriting the Cultural Heritage Framework for the Age of (Digital) Ecologies

Author: Jonatan Jalle Steller[1]

[1] *Academy of Sciences and Literature Mainz*

Corresponding Author: jonatan.steller@adwmainz.de

The Academy of Sciences and Literature Mainz (AdW) produces a number of critical editions of cultural heritage, which become increasingly multimodal over long funding periods of 12 or more years. Historical dictionaries, for example, turn to include geodata and statistics, image archives gain contextualisations and 3D reconstructions, and letter collections require annotations as well as digitised archive records. In addition, the Linked Open Data paradigm defines common formats to make (and keep) the editions' content available in, and federated search providers specify their additional APIs to support. Last but not least, digital editions should be accessible by default rather than as an exception.

To address such challenges, the AdW has tried to produce a fixed set of edition-related extensions for the content management system TYPO3 for several years now –with limited success. They were spread across multiple versions of the platform, hard to combine, and difficult to maintain as they were designed for one project and then heavily adapted for another without further documentation. In this talk I will outline the process of redesigning and rewriting this software stack.

The Cultural Heritage Framework 2 (CHF) is a toolkit for web apps that enable users to edit and publish cultural heritage data. Instead of abstract user stories, which have proven useful for the development of individual editions, the CHF was rebuilt based on media ecology theory. In this process, the software is seen as an entity in relation with various other human and non-human entities which can be grouped into media ecosystems. The product needs to be designed in a way that allows it to 'survive'and be seen as a good-faith actor in various such ecosystems, including (but not limited to) academic editorial teams, frequently changing and often inexperienced maintainers, mobile web browsers, content aggregators, and users with accessibility needs.

Compared to previous attempts, this analysis led to a different feature set including an adaptable and atomic user interface, embedded JSON-LD metadata, semantic main classes, standardised search functionality, an import/export mechanism, integrated user documentation, and extensive developer documentation. Through interlocking and coherent components for specific data types, projects using the software may now add functionality as they grow or change focus over time. All components feature interfaces to edit the data by reusing features of the TYPO3 platform, but also allow importing data from other systems that a project may use or exporting the data to TEI XML or triple stores.

The talk focuses on the practical application of media ecology theory in the CHF, which is not yet common in Digital Humanities software. It specifically dives into the consequences of evaluating accessibility software, its users, and entities with more limited requirements as one ecosystem instead of a single user story: accessibility needs to be observed not just on the level of content consumption but also on the levels of data analysis and production.

Slot length:

Research Software for Computing and Visualising Text / 113

TextAPI and TIDO - An example for research software product development

Author: Mathias Göbel[1]

Co-authors: Michelle Weidling [1]; Paul Pestov [1]

[1] *SUB Göttingen*

Corresponding Authors: goebel@sub.uni-goettingen.de, pestov@sub.uni-goettingen.de, weidling@sub.uni-goettinge.de

Introduction

Interoperability is key. Libraries, archives, and other repositories providing data to the public utilize a variety of metadata standards and interface specifications. TextAPI is a newly developed API specification (so far mainly used for digital scholarly editions) that provides metadata about textual resources while TIDO (Text vIewer for Digital Objects) is an application to display them. Our contribution will give a short introduction about this couple.

Providing (meta)data for digital editions via API makes applications using this data agnostic to the back end technology. This can significantly contribute to the sustainability of the project and the resulting architecture. TextAPI does not require any database at all, since a static copy of the resources provided by a webserver is sufficient for most features.

TextAPI

Features By referencing **Web Annotations** TextAPI resources can be extended to any scale. Currently we use Web Annotations to reference entities, highlight editorial desicions and other annotations.
The API is organized in a **modular** and **extensible** way to extend the core specifications with additional data for specific use cases.
The specification conforms to JSON-LD format to become part of the semantic web.

TIDO

TIDO is a web application based on VueJS operating in common with the TextAPI. It provides views to the data organized in panels and tabs. These views contains image, text, metadata, collection objects and annotations. It is designed to be a general-purpose front-end application to display TextAPI resources. Project implementators can rely on highly dynamic configuration options in order to assembly the desired presentation of their data. It is possible to define a strict panel setup (order and amount of panels and their content) or rely on TIDO's default settings to achieve quick results. The viewer enables researchers to interact with the text by displaying various types of annotations, for example editorial comments or references to external registers. That combined, users are able to exchange their current state of the application by sharing a citable URL. The TIDO bundle provides a lightweight solution that can be self-hosted on any web server architecture.

Roadmap

The architecture is not feature-complete, yet. Our requirements mainly derived from projects creating digital editions (e.g. transcriptions) include the display of multiple sources at once for comparing and analyzing multiple texts. Also the interactions between text, annotations and corresponding images has to be improved to complete our construction kit for digital editions.

When applied on many resources, TextAPI will allow for a central search index for all these resources and so –again - helps to scale vertically.

- centralized search index based on many TextAPIs
- in the future: building LLM upon many TextAPIs
- Variants: allow to reference variants of a certain text part in the annotation panel
- Text Comparison: compare and analyze multiple texts side-by-side

Slot length:

RSE Research / 114

What do GitHub repositories of Potsdam researchers tell us about quality and reproducibility of their scientific software?

Authors: Akshay Devkate[None]; Anna-Lena Lamprecht[None]

Corresponding Author: akshay.devkate@uni-potsdam.de

To enable and ensure the reproducibility and traceability of scientific claims, it is essential that scientific publications, the corresponding datasets, and the data analysies code are made publicly available.The adoption of the FAIR4RS principles and software engineering best practices could significantly enhance the success of delivering a codebase that produces consistent, reproducible results. Yet, not much is known about how practices like version control, continuous integration, scientific data management, automated testing and software documentation and citation are adopted within scientific community.
To gain a better understanding of the standards and practices followed by research software developers in Potsdam, we identified GitHub users who are PhD candidates, PostDocs, researchers or other academics affiliated with the University of Potsdam or one of the local research institutes. This focus is motivated by our goal of fostering collaboration and engagement with the research software development community in Potsdam. We then collected 13000+ open-source repositories of those users for thorough study to access coding practices and standards followed by them. The repositories collection has significant number of projects that are data analysis and web development with frequent mentions of keywords like data, web, machine learning along with use of programming languages like python and javascript. There is also a notable presence of non-technical projects, including educational repositories identified by terms like course, teaching, and workshop, along with other repositories with keywords survey, study, and assignments, thesis. To focus only on research repositories which are actually research software we classified them according to the DLRs application classes . This helped us to eliminate lot of repositories which do not contain relevant artefact and which are not meant for generating reproducible research results. We found 2100+ projects which falls under DLR application class 0 repositories which does not have relevant artefact that aligns with research software. In our study, which is presented as a short talk, we want continue to classify all open-source projects from the data collected into the DLR application classes with the goal of carefully assessing the use of git, git work flows, automated testing, and how they organize their code, test files, and documentation. Additionally, our goal is to assess the level of accessibility and quality of scientific documentation, code commenting, that ensure software reproducibility. With this short talk we hope to spark a discussion and further collaboration on this topic.

Slot length:

115

A place for research software in Helmholtz and beyond: the Helmholtz Research Software Directory

Authors: Christian Meeßen[1]; Felix Mühlbauer[1]; Martin Hammitzsch[1]; Uwe Konrad[2]

Co-author: Beate Hetenyi [3]

[1] *Helmholtz Zentrum Potsdam GFZ Deutsches GeoForschungsZentrum*

[2] *Helmholtz Zentrum Dresden-Rossendorf*

[3] *GFZ eScience Department 5.2*

Corresponding Author: christian.meessen@gfz-potsdam.de

Abstract: The Helmholtz Research Software Directory (Helmholtz RSD, https://helmholtz.software) is a platform designed for promoting and discovering research software. The platform has been launched in March 2023 by the Helmholtz Federated IT Services (HIFIS) to provide a platform for research software developed within Helmholtz. The Helmholtz RSD is based on the RSD of the Netherlands eScience Centre (https://www.esciencecenter.nl/), and allows to comfortably add, manage and track metrics of research software. In this demo session, we will learn how to build the RSD from scratch and discover the functionality of the RSD from an admin and end user perspective.

Slot length:

Workshop (1h)

Cross-Platform Development with C/C++ / 116

Controlling parallel simulations in Julia from C/C++/Fortran programs with libtrixi

Author: Michael Schlottke-Lakemper[1]

Co-author: Benedict Geihe [2]

[1] *RWTH Aachen University*

[2] *University of Cologne*

Corresponding Author: m.schlottke-lakemper@acom.rwth-aachen.de

The Julia programming language aims to provide a modern approach to scientific high-performance computing by combining a high-level, accessible syntax with the runtime efficiency of traditional compiled languages. Due to its native ability to call C and Fortran functions, Julia often acts as a glue code in multi-language projects, enabling the reuse of existing libraries implemented in C/Fortran. With the software library libtrixi, we reverse this workflow: It allows one to control Trixi.jl, a complex Julia package for parallel, adaptive numerical simulations, from a main program written in C/C++/Fortran. In this talk, we will present the overall design of libtrixi, show some of the challenges we had to overcome, and discuss continuing limitations. Furthermore, we will provide some insights into the Julia C API and into the PackageCompiler.jl project for static compilation of Julia code. Besides the implications for our specific use case, these experiences can serve as a foundation for other projects that aim to integrate Julia-based libraries into existing code environments, opening up new avenues for sustainable software workflows.

Slot length:

Local RSE-related Organisations / 117

House of Computing and Data Science

Author: Katrin Schoening-Stierand[1]

[1] *Universität Hamburg*

Corresponding Author: katrin.schoening-stierand@uni-hamburg.de

As a central institution of the University of Hamburg, the House of Computing and Data Science (HCDS) supports interdisciplinary research and application of innovative digital methods in close cooperation with its partners from science and research in the Hamburg metropolitan region.
It coordinates and supports the implementation of the digital strategy in research at the University of Hamburg. It fuels the easy adoption, usage, and research of digital methods in its Methodology Competence Center and offers various disciplines and projects a forum for the exchange of information and collaboration at the interface between methodological sciences and applied sciences in the Cross-Disciplinary Labs (CDLs).
In Cross-Disciplinary Labs, interdisciplinarity takes place at eye level: here, research questions are addressed that are of interest to both methodological science and its application in research in science and the humanities.

Slot length:

Continuous Integration - Advanced / **118**

Efficient Docker Container Management and CI Strategies for Research Software Engineering

Author: Dominik Thoennes[1]

Co-author: Harald Köstler [2]

[1] *Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU)*

[2] *Friedrich-Alexander Universität Erlangen-Nürnberg*

Corresponding Author: dominik.thoennes@fau.de

This presentation focuses on the continuous integration (CI) infrastructure used by two simulation software frameworks with a strong emphasis on performance and scalability: HyTeG, a C++ framework for large-scale high-performance finite element simulations based on geometric multigrid, and waLBerla, a massively parallel framework for multiphysics applications.

During this talk, we will detail our approach to managing Docker Containers within our CI environment. As we develop software for research purposes, we often face the challenge of testing various combinations of software packages, especially different compilers. To tackle this, we employ Python scripting and the Spack software manager to create Docker Containers. These containers are then utilized within the CI for our research software. The container creation process is automated within the CI itself, enhancing consistency and manageability.

Furthermore, we will explain how Python scripting generates the CI specifications for our GitLab CI. With approximately 150 different jobs in our Pipeline, a generated CI significantly reduces the chances of errors compared to manual creation and maintenance. Moreover, we will shine a light on some of the less conventional uses of CI, such as analyzing build times or conducting code style checks.

Our goal for this talk is to share best practices and spread ideas about more efficient CI utilization.

Slot length:

other(help with comment)

Policies and Legal Aspects / **119**

Research Software Policy Establishment at Helmholtz - Activities and Results

Authors: Tobias Schlauch[1]; Uwe Konrad[2]

[1] *DLR / HIFIS*

[2] *Helmholtz Zentrum Dresden-Rossendorf*

Corresponding Author: tobias.schlauch@dlr.de

The Helmholtz Association is a pioneer in the establishment of research software guidelines and policies in the German research landscape. The roots go back to one of the first German RSE focused workshops, which took place in Dresden in 2016. Since then, the field of RSE has been successively expanded at various levels through the provision of training and support services, technical platforms and, last but not least, the development of guidelines and policies. In context of research data management, a similar process has been driven with a strong focus on research data policies and data

management plans. Guidelines for software development are just as important in modern research, but have hardly been established to date.

The talk is a progress report on the activities and results that have been achieved in the Helmholtz Centers in recent years. We present concrete examples with facts, statistics and user experience reports. In addition, we also share our experiences on how to actively stimulate this process, for example, through awards and visible indicators.The policy implementation at Helmholtz is ongoing and is actively supported in regular Helmholtz-wide research software forums organized by the Helmholtz Incubator Platform HIFIS and the Task Group Research Software of the Helmholtz Open Science Working Group.

Slot length:

RSE Practices / **120**

Research Software Engineering at Intel

Author: Christopher Dahnken[1]

[1] *Intel Corporation*

Corresponding Author: christopher.dahnken@intel.com

There are no RSEs at Intel - or are there? Despite being a hardware manufacturer, Intel has one of the biggest software workforces in the industry. Apart from pure development, many engineers also perform an interface role between science and software that might well be characterized as research software engineering. We provide an overview of the work and impact of RSEs within Intel and in the broader ecosystem, and discuss the skill sets required and careers that can be pursued.

Slot length:

Poster Session / **121**

Poster - Intel

Author: Christopher Dahnken[1]

[1] *Intel Corporation*

Corresponding Author: christopher.dahnken@intel.com

We provide an overview of recent scientific results in HPC and AI developed by Intel's European Technical Computing Engineering team and partners in academia and industry.

Slot length:

Poster Session / **122**

FuncScan: A pipeline to mine DNA for antimicrobial peptide genes, antibiotic resistant genes, and biosynthetic gene clusters under one umbrella.

Author: Anan Ibrahim[1]

[1] *Paleobiotechnology Department, Leibniz-Institut für Naturstoff-Forschung und Infektionsbiologie e. V. Hans-Knöll-Institut, Jena*

Corresponding Author: anan.ibrahim@hki-jena.de

Over the past decade, major advances in the field of microbial biotechnology, in particular (eco)genomics have been observed. Because of the increase in genetic data produced at high throughput by next generation sequencing technologies, many bioinformatic tools for their *de novo* and reference-based assembly were and are continually being developed. In order to meet the strong demand for *accurate* and *ideal* assembly into DNA fragments, *i.e.* contigs, many standardized analysis pipelines have been created. This is even more complex in the case where these contigs must be compiled within metagenomes.

However, the diverse functions and biosynthetic capabilities of these ever-growing biomes remain largely elusive. The study of this functional diversity, however, is of utmost importance as it plays key roles in cell signaling, communication, microbial evolution, and defense. Many excellent software tools have been developed to predict antimicrobial resistance genes (ARGs), biosythetic gene clusters (BGCs) and antimicrobial peptide genes (AMPs). However, the use of such standalone tools requires a high minimum level of command line expertise from researchers in the fields of biology and chemistry. Many of these tools are written in different coding languages, lack reproducibility due to differing computing environments required and the results obtained are generated in different formats. This demands a steep learning curve for the researcher, eventually reducing time efficiency prior to downstream processing in the lab.

Here, we address this problem by creating a pipeline called **FuncScan** that can screen nucleotide sequences for functional genes. **FuncScan** is built using a specialized workflow management system called nf-core, which supports FAIR (findable, accessible, interoperable, and reusable) research methods. The pipeline consists of three autonomous workflows that can detect ARGS, BGCs, and AMPs, which can be activated using one
command. Each workflow combines different stand-alone tools that are compiled in independent software packages to resolve different execution logics and software dependencies. The standardized result of these workflows is then parsed and further filtered by newly developed parsing tools and visualized using associated Application Programming Interface (API). This allows for the easy comparison of the candidate genes predicted and removes all potential spurious or false positive genes. Therefore, FuncScan exponentially reduces the time and effort required by researchers in the natural product discovery field to identify genes that may be suitable for downstream processing in the lab.

Here, we use ancient DNA extracted from historical dental plaque as case study to explore the presence of *true* antimicrobial peptide genes that can be processed in the lab to generate potentially new AMPs.

Slot length:

Workshop (1h)

Poster Session / 123

Poster de-RSE e.V.

Authors: Frank Loeffler[1]; Jan Linxweiler[2]; Stephan Janosch[3]

[1] *Friedrich Schiller University Jena*

[2] *Technische Universität Braunschweig*

[3] *MPI-CBG*

Corresponding Author: j.linxweiler@tu-braunschweig.de

Here you can get into contact with people representing the de-RSE society.

Slot length:

Poster Session / **124**

FSFE Poster

Author: Michele Martone[1]

[1] *Leibniz Supercomputing Centre*

Corresponding Author: michele.martone@lrz.de

What makes software “free”?
What software make us “free”?
And who makes free software?
Can I make software free without making free software?
For a chat about anything free software, come at the stand with material from the Free Software Foundation Europe.

Slot length:

Poster Session / **125**

Real-Time Data Visualisation in Experiments Using a Generalised Asynchronous Live Plotting Module, a Python Example

Author: Mingsong Wu[1]

[1] *University of Geneva*

Corresponding Author: mingsong.wu@unige.ch

Automating measurements is an increasingly important skill in experimental physics for laborious and repetitive tasks. A typical automation process uses Python to create interfacing modules for controlling and reading instruments. However, even if the modules are experiment-agnostic, the top-level experimental control scripts are often highly specialised, acting as instruction sequences instead of an interactive instrument console. Where real-time data visualisation is needed, the control scripts become complicated quickly, with graphical user interfaces (GUI) that are difficult to adapt for different setups. To address this problem, we introduce a drop-in live plotting Python module with a generalised input format; a “data logger”control script uses it to generate asynchronous live plot windows while still controlling instruments via simple interactive shell function calls. If desired, a separate GUI wrapper can then be applied on top of the data logger. We demonstrate this scheme of work in the context of single-photon source experiments and show its versatility with different experimental configurations.

Slot length:

Poster Session / **126**

KONWIHR

Authors: Florian Goth[1]; Gerasimos Chourdakis[2]

[1] *Universität Würzburg*

[2] *Technical University of Munich*

Corresponding Author: gerasimos.chourdakis@tum.de

The KONWIHR Poster

Slot length:

Poster Session / 127

System regression tests for the preCICE partitioned coupling ecosystem

Authors: Benjamin Uekermann[1]; Gerasimos Chourdakis[2]; Valentin Seitz[2]

[1] *University of Stuttgart*

[2] *Technical University of Munich*

Corresponding Author: valentin.seitz@tum.de

While Unit and Integration testing provide quick insights into individual software components, ensuring seamless collaboration among all parts necessitates system-level testing. In the context of preCICE, a coupling library enabling plug-and-play partitioned multi-physics simulations, unique challenges arise. The coupling library stands at the center of a rich coupling ecosystem, with components organized in different repositories, and with complete simulations only possible by combining multiple components from different layers. This additional complexity, together with the heterogeneity of generated results by different simulation software that is coupled together, poses unusual challenges.

The preCICE system tests allow developers from different perspectives (library developer, adapter developer, example cases contributor, release manager) to verify that all components still work together, and that complete simulations still give the same results over time.

A collection of simple Python scripts reads test definitions defined in YAML-based configuration files. Tests describe which combinations of which versions of components to use to start a simulation, extract comparable result files, and compare them. Behind the scenes, the framework starts Docker containers as Docker Compose services, and compares results with fieldcompare.

The tests can be executed by any developer on their local system, or through the GitHub Actions infrastructure of preCICE, on a dedicated organization-wide runner. The workflow can be executed manually, or from different repositories, combining versions that correspond to the perspective of each developer. Examples include combining a feature branch with other released branches, or all development branches of each component. Test reports are generated as build artifacts, archived and easily discoverable next to the GitHub Actions logs.

This poster will present the challenges that testing such a coupling ecosystem brings, and will present a now complete and working solution for system and regression testing in preCICE.

Slot length:

Lessons learned and applied / 128

Refactoring a research code for multi-physics simulations: the 4C experience

Author: Georg Hammerl[1]

Co-authors: Sven Berger [1]; Sebastian D. Proell [2]; Matthias Mayr [3]; Niklas Neher [4]; Erik Faulhaber [5]; Ingo Scheider [1]; Hoang-Giang Bui [1]; Martin Kronbichler [6]; Alexander Popp [3]; Wolfgang A. Wall [2]; Daniel Hoeche [1]; Christian J. Cyron [1]

[1] *Helmholtz-Zentrum Hereon*

[2] *Technical University of Munich*

[3] *University of the Bundeswehr Munich*

[4] *University Stuttgart*

[5] *University of Cologne*

[6] *University of Augsburg*

Corresponding Author: georg.hammerl@hereon.de

Research software often starts with small codes that expand over the years and by the work of various people into large codes. This evolution is shaped by the specific research projects for which codes are used and the pressure to achieve within these research projects milestones and publications. This pressure is often counteracting the aim to develop a rigorous and clear structure for the underlying research codes. In addition, research codes are typically not developed by professional software engineers but rather by PhD students and postdocs without special training in software engineering. All this inevitably leads to large legacy codes that become increasingly difficult to handle.

As an example of such problems, we discuss the multi-physics simulation code 4C which consists of more than 1 million lines of code and has been developed over 20 years by dozens of researchers in collaboration of several institutions. 4C is a parallelized multi-physics research code for analyzing and solving a plethora of physical problems with focus on computational mechanics. It offers simulation capabilities for various physical models, including single fields such as solids and structures, fluids, scalar transport, or porous media, as well as multi-physics coupling and interactions within multi-field problems.

To overcome the initially summarized difficulties, we applied two strategies. On the one hand, we started new code projects based on the advanced library deal.II and on the Julia project Trixi.jl and sought to port the functionalities of 4C into these new codes. On the other hand, we pursued incremental refactoring of the existing legacy code. In this talk, we compare the pros and cons as well as the results of both approaches in the case of 4C.

Slot length:

129

Keynote: Demystifying Entropy

Author: Haye Hinrichsen[1]

[1] *Uni Würzburg*

Corresponding Author: haye.hinrichsen@uni-wuerzburg.de

The term entropy was originally introduced by the physicist Rudolf Clausius as a quantity which describes the ability of a physical system to change its state in a thermodynamic process. But at least since the pioneering work of the mathematician Claude Shannon, entropy has also become a central concept in information theory. How are these two interpretations related? What exactly is entropy and how can we use it to understand thermodynamic quantities such as temperature? This lecture

aims to provide an introductory overview of these questions in the area between computer science and physics. One focus will be the numerical estimation of entropy through sampling. As with any nonlinear estimation measure, systematic errors occur when estimating entropy, but these errors can be significantly reduced using suitable mathematical methods. The lecture also deals with the question of how these correction procedures can be implemented algorithmically.

Slot length:

130

Keynote: Between innovation and regulation - requirements on research software in the biomedical domain

Corresponding Author: dagmar.krefting@med.uni-goettingen.de

Contributing to better health has been the motivation for many software developers to stay in academia despite excellent job perspectives in industry. However, as soon as digital health data is processed with your software, things get easily complicated today. You still can download open health data and play around, but as soon as it is getting close to patient care and you really want to have impact to health practice, you find yourself inbetween privacy risk and software quality assessments, software validation and liability risks. You need to be compliant to the medical device regulation, and likely the AI and the cyber security act –and of course the general data protection regulation with its various local dialects (aka Landesdatenschutzgesetze). This talk will give you an idea on the complexity of biomedical software engineering – and why open source software should be an ethical mandate in this field.

Slot length:

Poster Session / **131**

Experience Report: How to tap the minds of the brightest students for RSE?

Authors: Florian Angermeir[None]; Michael Dorner[1]

Co-authors: Daniel Mendez [1]; Davide Fucci [1]

[1] *Blekinge Institute of Technology*

Corresponding Authors: michael.dorner@bth.se, angermeir@fortiss.org

In this poster, we report our experiences integrating research software engineering into a software testing course at Blekinge Institute of Technology, Sweden. In groups of four to five students, the teams implemented a comprehensive test suite entailing broad basic tests with an additional test-specific focus (e.g. performance testing, algorithmic verification) for real world research software. This integration resulted in a comprehensive test suite for the research software, significantly improved documentation, minimized code dependencies, and maximized code portability. The project brought students closer to SE research and was well received. However, there is no free lunch: It required significant effort, and integrating student code into open-source projects was impossible because of the uncertainty about the intellectual properties of code contributions from students.

Slot length:

Poster Session / 132

MEiNunG: Analysing Moral Stances in Text

Authors: Christian Nix[1]; Julius Mankau[1]; Maximilian Josef Frank[1]; Nikola Martinov Staykov[1]; Tim Knothe[1]; Viktoria Obermeier[1]; Yunqing Wang[1]

[1] *Technical University of Munich*

Corresponding Authors: tim.knothe@tum.de, christian.nix@tum.de, maximilian.j.frank@tum.de, viktoria.obermeier@tum.de, yunqing.wang@tum.de, nikola.staykov@tum.de, julius.mankau@tum.de

Ever since humans have been able to speak, words have been a tool for conveying what is important to us, what values we hold and what opinions we stand for. At the intersection of technology and ethics, we use machine learning to decode moral principles expressed in the language we use. However, quantifying ethical or moral views in media e.g. newspaper articles, interviews and books remains difficult until today. Machine Learning algorithms and AI evolve fast, making it possible to examine and evaluate huge amounts of text for underlying patterns. There are also different rating instruments for moral perspectives such as the Moral Foundations Dictionary. We aim to merge the fields of computer science and philosophy in a very interdisciplinary team. The development of a machine-based learning algorithm allows us to understand the moral connotations and views in thousands of e.g. newspaper articles at a time. Our goal is to use this technology to extract moral stances from textual sources, thereby gaining deeper and more nuanced insights into the societal evolution of ethical perspectives. Talking about euthanasia or assisted suicide: even though both terms explain the same issue, the moral valuation is entirely different. By observing ethical shifts and the transformation of opinions, we try to extrapolate the trend for the future. Our software and analysis methods provide helpful insights for connotation research of texts also in other areas e.g. political, sociological or other research. We also aim to guide meaningful discussions, promote understanding, and foster a informed and morally conscious future –e.g. by telling which articles might be biased.

Slot length:

Poster Session / 133

Advancing Digital Transformation in Materials Science: Scientific Workflows within The Platform MaterialDigital

Authors: Joerg Schaarschmidt[1]; Tilmann Hickel[2]

Co-authors: Alexander Straumal [1]; Artem Buldin [1]; Celso Rego [1]; Wolfgang Wenzel [1]

[1] *Karlsruher Institute of Technology*

[2] *Bundesanstalt für Materialforschung und -prüfung*

Corresponding Author: joerg.schaarschmidt@kit.edu

The ‘Platform MaterialDigital’ (PMD), supported by the German Federal Ministry of Education and Research (BMBF), is spearheading efforts to bundle and coordinate digitalisation within the field of materials science and engineering. Central to its mission are the deployment of decentralised data servers, the adoption of uniform data formats, and the implementation of digital processes through scientific workflows.

As the first funding phase draws to a close, PMD is at a crucial juncture. The groundwork, laid predominantly through academic research, has established a solid foundation. The transition to the second funding phase, now under the leadership of industry-led projects, seeks not only to broaden PMD’s impact within the materials science community but also to evaluate the practicality of the developed solutions.

At the heart of PMD's digitalisation strategy are the workflow frameworks pyiron and SimStack, highlighting the platform's commitment to scientific workflows. This presentation will delve into notable advancements within this area, including the integration of semantic data and the semantic characterization of workflows. Furthermore, an overview of the Workflow Store's current status and future trajectory will be provided. This central repository, which offers guidelines for consistent standards in scientific workflows and their components, plays a crucial role in disseminating established workflows and promoting standardization across the community.

Slot length:

Poster Session / **134**

hypertiling - Hyperbolic Lattices for Everyone!

Authors: Florian Goth[1]; Manuel Schrauth[None]; Yanick Thurn[2]

[1] *Universität Würzburg*

[2] *Julius-Maximilians Universität Würzburg*

Corresponding Authors: florian.goth@physik.uni-wuerzburg.de, yanick.thurn@uni-wuerzburg.de, manuel.schrauth@iis.fraunhofer

hypertiling (hypertiling.de) is a high-performance Python 3 library for the generation and visualization of hyperbolic tilings, embedded in the Poincaré disk model. Using efficient, precise and robust algorithms, computational lattices with millions of cells can be created in a matter of minutes on a single computer and are ready to be used for all sorts of scientific as well as artistic purposes. Hyperbolic geometry, i.e. negatively curved space, is a rapidly growing field of interest in physics and beyond. While this puts this formidable package into a sweet spot for future growth, we are left with a challenge: how to start making it known to potential users?

In this poster we detail a carefully crafted strategy that hopefully enables us to reach beyond our own scientific domain. In order to achieve this we use established open source practices, and collaborative development on GitLab. On top of that we use the potential of modern social media platforms, like YouTube, Twitter and Instagram to build a community across fields.

Slot length:

135

Closing Session

Corresponding Author: florian.goth@physik.uni-wuerzburg.de