The difference between RSE and Data Science

Firstname1 Lastname1 [1,2], Firstname2 Lastname2 [2], Firstname3 Lastname3 [3] und Firstname4 Lastname4 [1]

Abstract: Die LaTeX-Klasse `lni` setzt die Layout-Vorgaben für Beiträge in LNI Konferenzbänden um. Dieses Dokument beschreibt ihre Verwendung und ist ein Beispiel für die entsprechende Darstellung.

Keywords: LNI Guidelines, LaTeX Vorlage

1 What has been discussed

When discussing competences for research software engineers as the basis of a curriculum to train Research Software engineers (RSE) professionals it is logical to look at the data science movement for inspiration. Similar to RSE, Data Science (DS) is a cross-cutting field that focuses on a special area within the sciences or data oriented businesses.

The major computing organizations ACM and IEEE published a joint recommended computing curriculum which already included data science: the cc2020 Computing Curriculum already lists DS as one of the special cases for computing competences. Whilst DS is mentioned, research software engineering was absent.

Even though it is intuitively clear that RSE is not a subclass of DS, there have been few attempts of differentiating the fields with regard to their competency sets.

TODO insert previous research to RSE and DS: Levels of Differences between RSE and DS:

- institutional
- disciplinary connections
- target groups
- political history

[1] Universität 1, Abteilung, Straße, Postleitzahl Ort, Land,
firstname1.lastname1@affiliation1.org, https://orcid.org/0000-0000-0000-0000;
firstname4.lastname4@affiliation1.org, https://orcid.org/0000-0000-0000-0000

[2] University 2 , Department, Address, Country,
firstname1.lastname1@affiliation1.org, https://orcid.org/0000-0000-0000-0000;
firstname2.lastname2@affiliation2.org, https://orcid.org/0000-0000-0000-0000

[3] University 3, Department, Address, Country,
firstname3.lastname3@affiliation1.org, https://orcid.org/0000-0000-0000-0000

2 RSE and DS embeddings in the Research Cycle

Both RSE and DS can be conceptualized as a cross-cutting concern in many disciplines. However, the definition and relevance of these issues can be generalized based on the function they fulfill in the research cycle Figure 1.

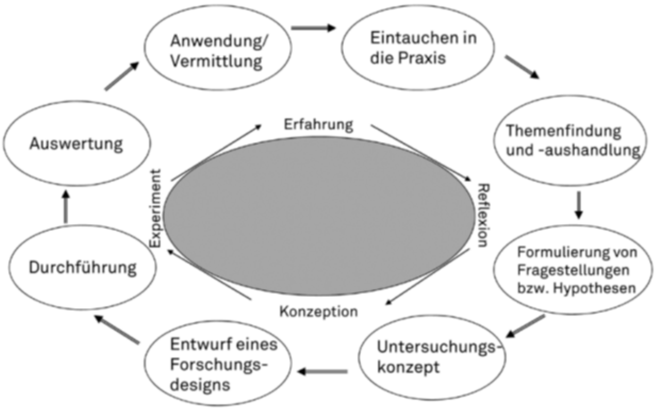

Fig. 1: The research cycle [Wi09] integrates the typical research process with the learning process.

There are different research processes depending on the discipline and the research question. However, [De21] showed that most of the research processes contain the following phases:

1. conceptualization (developing research questions, concepts)
2. design (developing the tools, instruments and concrete process models)
3. implementation (executing the experiment, study)
4. analysis & interpretation
5. dissemination (publishing, distributing, peer-review)
6. reflexion and improvements

For example, in the case of the field of learning technologies, the design phase often consists of extensive software development of different tools for learning. In this situation developing complex tools for learning can be considered as research software engineering. The analysis of what learners can gain from using these technologies can be conceived as an educational research in its own right. This example highlights the differences of scale in both the weight of the different process phases (here: phase 2) and the relevance of research engineering. It also shows a situation where research software engineering clearly differs from data science. A typical data science background would not enable researchers to build full-stack software that solves inefficiencies or hard-to-teach problems in education.

In the second example a GPT-like attention model is trained to classify data gained from the James-Webb telescope. Due to vast amounts of data and the continuous stream of new data research software engineering is needed to implement a pipeline for data cleaning, data warehousing and in-time analysis. In this case, the analysis & interpretation phase (4) has much more relevance. Another point of this is example is that data science competencies such as vectorization of algorithms, statistical analysis, machine learning etc. are interconnected with competencies from software engineering such as software architectures, software project management, and database programming. In this case the distinction between DS competencies and RSE competencies is very fluid.

The main argument behind these examples is that data science and research software engineering have a lot in common in terms of software development for science but show major differences where they are placed in the research process. This reframes the questions like "how much programming is in DS" or "how much engineering is in RSE" to a more structured approach of which cross-cutting functions exist in the research cycle that require computational means and which functions are part of the core identity of data scientists or research software engineers.

As a working hypothesis based on the example above and experience in the field we assume the following: [H1] RSE focuses more on concept development (if the research is computationally heavy), design, implementation and dissemination, i.e. phases 1,2,3,6 whereas DS focuses more on analysis, interpretation and dissemination (phases 4,5,6). Moreover, a second hypothesis would be that RSE often plays a role in shaping the context of the research [H2a], such as integrating projects with similar concerns, open source development and institutional needs. In contrast, data science is exclusively embedded in the research [H2b].

The focal point of the following chapter will revisiting existing ideas for DS and RSE curricula and map the competences outlined there to the phases in the research process. This should give the abstract discussion above empirical grounding and can be used to test the hypothesis.

3 Discussion

The lists of extracted DS and RSE competency clusters can be found in Appendix. We also plan to publish a full table of RSE competencies (and possible DS competencies) sorted by clusters. In terms of this top-level discussion the competency clusters are sufficient to compare the differences with regard to the research cycle.

H1 could not be confirmed in the strict sense. Although the compiled competency clusters show that there is a stronger focus on certain stages of the research process for both RSE and DS competencies, both can be interpreted more generally to encompass all stages of the research cycle. Moreover, there are some competency description that seem very similar such as the focus on the research cycle for RSE and the data science lifecycle for DS. In terms of methodology simply comparing the existing mentions of competencies should not be regarded as the best possible proxy to the actual distribution in the field. A survey study asking practitioners and researchers where in the process they would place DS and RSE would yield far more convincing results. Still, self-evaluation might also be biased depending on the identity of the people working the in the respective fields.

The most clear differences between DS and RSE are found in the design stage and the analysis stage. The design stage holds most of the competency clusters the RSE community defined. The DS counterpart is very general and many competencies listed there could in fact be construed as RSE-competencies that are imported for more complex cases. In contrast, the analysis stage is more connected to DS. This can be explained by the historic challenges software development faces in terms of clear-cut evaluation but also by the distribution of labor: if the RSE-job ends with the developed software and the core experiment or study uses the software as a tool, the analysis part is then handed over to the respective field specialists.

It is very hard to both evaluate the impact of technology and also to evaluate the technology itself and its impact on the study. Comprehensive methods like Directed Acyclic Graph Modellings (DAGs) or Instrumental Variables try to tackle these nested evaluation issues but have not found widespread use.

H2 also had to be rejected: in fact, DS seems to contain more aspects outside the research cycle than RSE. Even though the core analysis of data component is very embedded in research, DS has a lot of institutional, political and legal challenges. Research Data Management (RDM) could be named as the most prominent of these. Due to the strong overlap of non-research related competencies, a joint list competency clusters was compiled that lists the competences that are not research cycle related (also in the appendix).

The long list of transversal competencies begs the question if there are also technical competences that overlap. Even though this was not the focus of the analysis, these can be easily spotted by investigating the shift of data analysis to artificial intelligence based methods. Training, fine-tuning and mainstreaming large language models requires more and more computing power, stable infrastructure and network components. On the other hand, CPU-based software-engineering becomes less demanding and also profits from AI-generated algorithm and code development. However, not all software engineering boils down to the current AI-hype. In summary, there is no clear way of generalizing whether DS or RSE need more and deeper understanding of computer science.

References

[De21] Dehne, J.: Möglichkeiten und Limitationen der medialen Unterstützung forschenden Lernens, de, PhD thesis, 2021, https://publishup.uni-potsdam.de/49789.

[GAB+24] Goth, F.; Alves, R. S.; Braun, M., et al.: Foundational Competencies and Responsibilities of a Research Software Engineer [version 1; peer review: 2 approved]. F1000Research 13, p. 1429, 2024, https://doi.org/10.12688/f1000research.157778.1.

[Ge21] Gesellschaft für Informatik e.V. (GI): Empfehlungen für Masterstudiengänge „Data Science“ – auf Basis eines Bachelors in (Wirtschafts-)Informatik oder Mathematik, https://gi.de/fileadmin/GI/Hauptseite/Service/Publikationen/Empfehlungen/Empfehlungen_Masterstdiengaenge_DataScience_2021.pdf, Zugriff am 29. April 2025, 2021.

[Pe25] Petersen, B. et al.: Lernzielmatrix zum Themenbereich Forschungsdatenmanagement (FDM), version 3, 2025, https://doi.org/10.5281/zenodo.15025246.

[Wi09] Wildt, J.: Forschendes Lernen: Lernen im „Format “der Forschung. journal hochschuldidaktik 20 (2), pp. 4–7, 2009.

4 Appendix

4.1 Glossary

C A general-purpose programming language often used for system-level development.

Cpp C++ — an extension of C that supports object-oriented programming.

DIST Software distribution — the practice of packaging and delivering software and its dependencies.

DOCBB Documentation and best practices — ensuring code is understandable and maintainable.

DOMREP Domain repositories — platforms that store and share domain-specific research data.

HPC High-Performance Computing — using supercomputers and parallel processing for complex tasks.

MOD Modularity — the design principle of separating software into interchangeable, functional components.

NEW Novel research — work that contributes original insights to a scientific domain.

PM Project Management — planning, executing, and overseeing projects effectively.

Python A high-level programming language widely used in data science and scripting.

R A programming language and environment for statistical computing and graphics.

RSE Research Software Engineer — someone who applies software engineering skills to scientific research.

SP Software publication — the process of preparing and disseminating software artifacts.

SRU Software reuse — the practice of using existing software components in new projects.

STEM Science, Technology, Engineering, and Mathematics.

SWREPOS Software repositories — systems for storing and managing software code and versions.

TEAM Teamwork — the ability to collaborate effectively in a group setting.

TEACH Teaching — the skill of communicating knowledge and helping others learn.

USERS End users — the scientists or researchers who rely on software tools.

4.2 DS and RSE Competences in relation to the Research Cycle

In the following the contents of [Ge21] and [Pe25] are parsed as the most current examples of data science curricula in the German research context. The contents are inspected for obvious links to the research process and interpreted if no explicit connections are made.

For the RSE side the contents of [GAB+24] are used as a basis as well as the current state of the RSE-Curriculums project. For the RSE competencies the RSE community has developed short codes. These are attached in the glossary in the appendix.

In terms of methodology it should be noted that this approach follows a community-driven consensus building. It should not be mistaken for a review study with measurable intersubjectivity based on instruments like PRISMA.

1. Conceptualization

RSE Competencies:

- Understanding the research cycle (`RC`)
- Conducting and leading research (`NEW`)

Data Science Competencies:

- Understanding the Data Science Lifecycle and methods selection
- Awareness of data context, purpose, and interdisciplinary implications (ethics, economics, legal)

2. Design

RSE Competencies:

- Software Design
- Software re-use strategies (`SRU`)
- Creating documented code building blocks (`DOCBB`)
- Software behavior analysis (`MOD`)

- Building distributable software (`DIST`)
- Tool and environment configuration (`SWLC`, `SWREPOS`)
- Full-Stack Programming

Data Science Competencies:

- Data integration & feature engineering (ETL, pipelines, quality checks)
- Design of data workflows and modeling processes
- Visualization theory and editorial thinking (exploratory analysis)
- Algorithm selection and objective function definition (ML core)
- Designing responsible data usage frameworks (ethics, privacy)
- Data-oriented Programming (R, Python, Matlab etc.)

3. Implementation

RSE Competencies:

- Source control, testing, CI/CD (`SWREPOS`, `DIST`, `DOCBB`)
- Working in interdisciplinary teams (`TEAM`)
- Project and task management (`PM`)

Data Science Competencies:

- Deployment of data pipelines and operational models
- Tool usage (Python, R, Julia, ML libraries like scikit-learn, Dask)
- Executing experiments using machine learning, deep learning
- Applying project management and interdisciplinary communication

4. Analysis & Interpretation

RSE Competencies:

- Software behavior interpretation (`MOD`)
- Documentation of research results and workflows

Data Science Competencies:

- Explorative Data Analysis (EDA), multivariate visualizations
- Deep understanding of model inference and optimization strategies
- Time series analysis, pattern mining, and argumentation (Data Mining)
- Ethical analysis of bias, fairness, and model accountability

5. Dissemination

RSE Competencies:

- Software publication and citation (`SP`)

- Use of domain repositories (`DOMREP`)
- Teaching and communication (`TEACH`, `USERS`)

Data Science Competencies:

- Reporting results and dashboards
- FAIR principles and reproducibility practices
- Preparation of software and models for open science platforms
- Communicating findings across disciplinary and public boundaries

6. Reflexion and Improvements

RSE Competencies:

- Continuous integration and testing (`SWLC`, `MOD`)
- Feedback-informed iterative development
- Mentoring, community involvement, and ethics (`TEAM`, `USERS`)

Data Science Competencies:

- Critical reflection on model performance and bias
- Model tuning, reengineering, and lifecycle updates
- Reinforcement learning and emerging AI models
- Responsible innovation, economic awareness, data sovereignty
- Application of Data Science in real-world domain projects

4.3 DS and RSE competencies outside the Research Cycle

This section collects transversal competencies in Data Science and Research Software Engineering (RSE) that support research but are not tied to a specific phase of the research process.

1. Ethical and Legal Awareness

Data Ethics

- Awareness of bias and fairness in data
- Model transparency, explainability, and accountability
- Risk and impact assessment (e.g., algorithmic decision-making)
- Ethical reflection throughout the data lifecycle

Data Privacy and Legal Contexts

- National and international data protection laws

- Differentiation between personal and non-personal data
- Licensing and data usage regulations
- Data compliance skills

2. Interdisciplinary Collaboration

Communication and Project Skills

- Project planning and evaluation
- Communication across disciplines and with non-experts
- Knowledge of marketing and dissemination strategies

Domain Integration and Labs

- Ability to adapt and apply methods in various domains
- Independent and team-based domain project execution
- Research data management within domains
- Interdisciplinary labs as agile learning environments

3. Computer Science Specialties

Various CS specializations such as

- database programming
- constraint programming
- machine learning
- algorithmic design
- . . .