Technology Research Software for Engineering Research

Wilhelm (Willi) Hasselbring

Software Engineering, Kiel University, Germany
https://se.informatik.uni-kiel.de

School of Electronics & Computer Science, University of Southampton, UK
https://www.southampton.ac.uk/people/5xcdkc/professor-wilhelm-hasselbring

TUM, January 22, 2025

C A U
Kiel University
Christian-Albrechts-Universität zu Kiel

University of Southampton

Research Software Engineering

Forschungssoftware effizient erstellen und dauerhaft erhalten

| LARS GRUNSKE | ANNA-LENA LAMPRECHT | WILHELM HASSELBRING | BERNHARD RUMPE | **Viele Forschungsprojekte an Universitäten sind ohne entsprechende Software nicht mehr denkbar. Software entwickelt sich zur relevanten Infrastruktur, die gepflegt, weiterentwickelt und gewartet werden muss. Mit Research Software Engineering (RSE) sollen geeignete Rahmenbedingungen geschaffen werden. Handlungsempfehlungen im Überblick.**

Der Begriff „Forschungssoftware" (engl. „research software") bezeichnet Software, die während des Forschungsprozesses oder für einen Forschungszweck erstellt wird. Forschungssoftware ist heute für viele wissenschaftliche Aktivitäten zwingend erforderlich. Sie kann zum Sammeln, Verarbeiten, Analysieren und Visualisieren von Daten, zur Erkennung von Zusammenhängen und zur Modellierung komplexer Phänomene und zur Durchführung anspruchsvoller Simulationen vom Material- über das Zell- und Organverhalten, soziale und ökonomische Beobachtungen, über das Wetter, das Klima der Erde bis hin zu Galaxienhaufen verwendet werden. Forschungssoftware spielt daher heute in fast allen Fächern eine entscheidende Rolle für die Forschung.

50 Jahre Software Engineering

Software Engineering (SE) hat sich in fast allen Universitäten und Fachhochschulen als eigenständiges Forschungsgebiet etabliert. Dabei haben die Professorinnen und Professoren durch ihre Forschung ein umfassendes Verständnis über die systematische und ingenieurtechnische Softwareentwicklung erarbeitet und dies nachhaltig in der Industrie etabliert. Dieses Wissen ist in Teilbereichen des Software Engineering wie etwa Anforderungsmanagement, Architektur, Design, Modellierung, Testen, Entwicklungsprozesse und angewandte formale Methoden organisiert, die sich weit über die Programmierung hinaus erstrecken.

Das Gebiet des Software Engineering entwickelt sich dennoch kontinuierlich weiter, weil neue Technologien neue Arten von Software und damit neue Herausforderungen für das Software Engineering mit sich bringen: Software ist sehr heterogen und reicht von eingebetteter Software und autonomen Steuerungen bis hin zu Desktop- und KI-Systemen, Geschäftssoftware und auch Forschungssoftware. Dabei sind die Probleme immer die gleichen:

- Wie lässt sich sicherstellen, dass die Software richtig und korrekt funktioniert?
- Wie kann die Qualität der Software sichergestellt werden?
- Wie lässt sich Software effizient entwickeln?
- Wie kann Software weiterentwickelt und langfristig nutzbar erhalten werden?
- Wie lassen sich Zeitvorgaben und Budgetbeschränkungen einhalten?
- Wie kann Software verallgemeinert werden, um mehr Nutzerinnen und Nutzer zu finden?

Die Lösungen und die sich daraus ergebenden Entwicklungstechniken sind in den verschiedenen Teilaktivitäten der Softwareentwicklung jedoch zumeist sehr unterschiedlich, denn unter anderem die Ausgangssituation, die Art der Software, die Komplexitätstreiber, die benötigten Qualitätsmerkmale, der Kontext, in dem die Software eingesetzt werden soll, sowie die regulatorischen Vorgaben unterscheiden sich stark. Die Software Engineering Community hat durch ihre Forschung schon viele Innovationen angeschoben, die oft auch breitere Nutzung finden. Dazu gehören zum Beispiel Wikis (die Grundlage der Wikipedia), agile Entwicklungsprozesse, Open Source (als Vorlage für Open Science) und eine Vielzahl an Werkzeugen zur Automatisierung in der Produktentwicklung, Informationsgewinnung mit Entwicklungs-Dashboards, für kollaborative Arbeitstechniken, für Versions- und Variantenmanagement und noch einiges mehr. Variantenmanagement mit Produktlinien, explizites Anforderungsmanagement und modellbasierte Entwick-

AUTOREN

Foto: privat

Lars Grunske ist Professor für Software Engineering an der Humboldt-Universität zu Berlin.

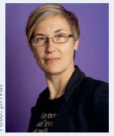


Foto: privat

Anna-Lena Lamprecht ist Professorin für Software Engineering an der Universität Potsdam.

Foto: privat

Wilhelm Hasselbring ist Professor für Software Engineering an der Universität zu Kiel.

Foto: privat

Bernhard Rumpe ist Professor für Software Engineering an der RWTH Aachen.

RSE Praxis
Bewährte Praktiken für die Entwicklung von Software im Forschungsalltag

RSE Training
Entwicklung von (R)SE-Fähigkeiten bei Forschenden und von R(SE)-Fähigkeiten bei Softwareentwickler/-innen

RSE Infrastruktur
Unterstützung bei Entwicklung, Betrieb und Wartung von Forschungssoftware

RSE Community

RSE Karrierepfade
Entwicklung von RSE als eigenes Berufsprofil und Karrierewegen für RSEs

RSE Interessenvertretung
für institutionelle Unterstützung, Finanzierung und Anerkennung von RSE und RSEs

RSE Forschung
Analyse und Verbesserung (des Entwicklungsprozesses) von Forschungssoftware

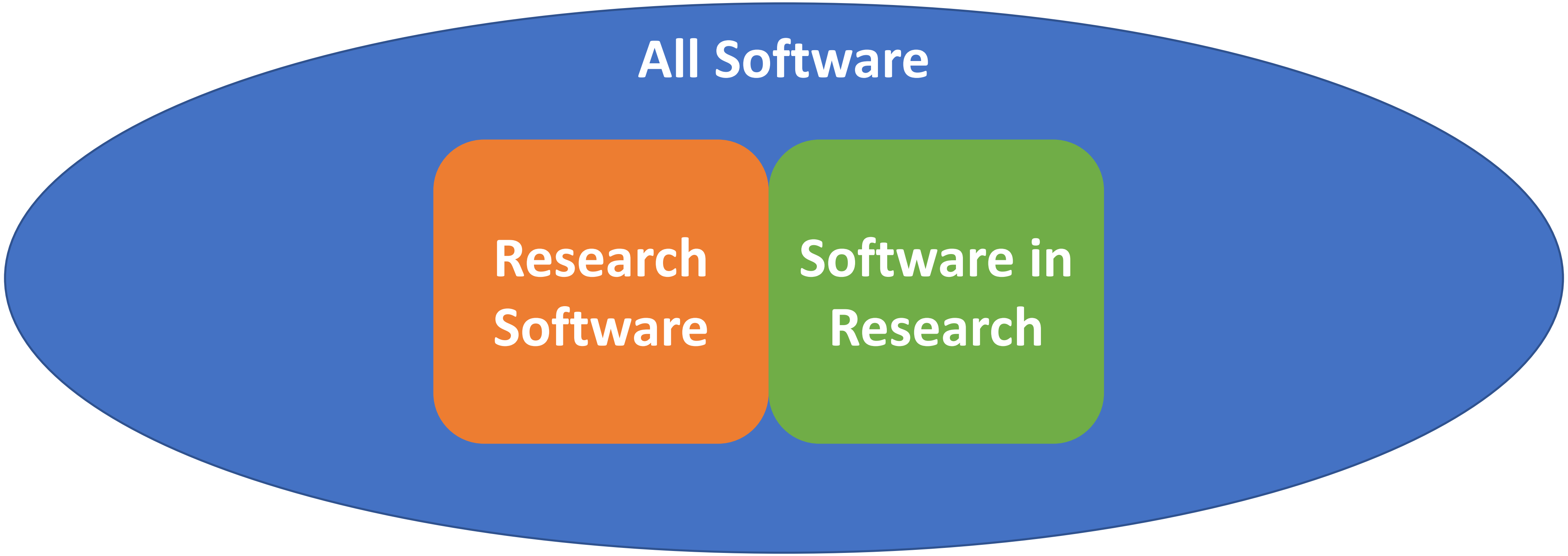
Research Software

RDA FAIR for Research Software (FAIR4RS) WG [Chue Hong et al. 2022] :

- **Research software** includes source code files, algorithms, scripts, computational workflows, and executables that are created **during** the research process or **for** a research purpose.
- Software components (e.g., operating systems, programming languages, libraries, etc.) that are used for research but were not created during or with a clear research intent should be considered `**software in research**´ and not `research software´.

Research software should be **FAIR** [Hasselbring et al. 2020b, Lamprecht et al. 2020] and **open** [Hasselbring et al. 2020a].

Software Segmentation



[Chue Hong et al. 2022]

Research Software
created during the research process or for a research purpose
Software in Research
used for research but not created during or with research intent

Context: German Special Interest Group GI-Fachgruppe “Research Software Engineering”

Interdisciplinary forum for:

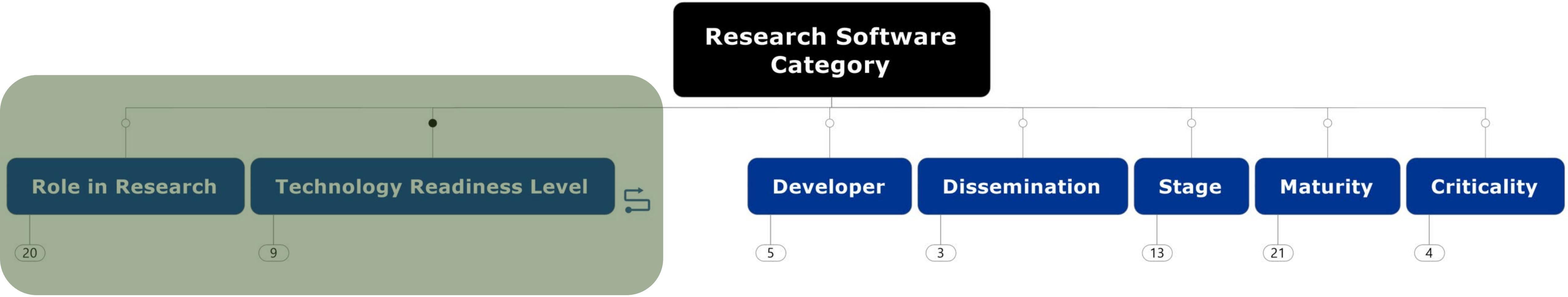
- Software Engineering Researchers
- Research Software Engineers

https://fg-rse.gi.de/ (German)

Task Forces (Arbeitskreise):

- Categories of Research Software
- RSE Advocacy Strategy
- RSE Community Events
- RSE Online Community
- RSE Research
- RSE Software Development Guidelines
- RSE State of Nation Report

Multi-Dimensional Categorization



[Hasselbring et al. 2024]

Roles of Research Software

Research software’s roles mainly fall into one of the following top-level role categories (and sometimes combinations):

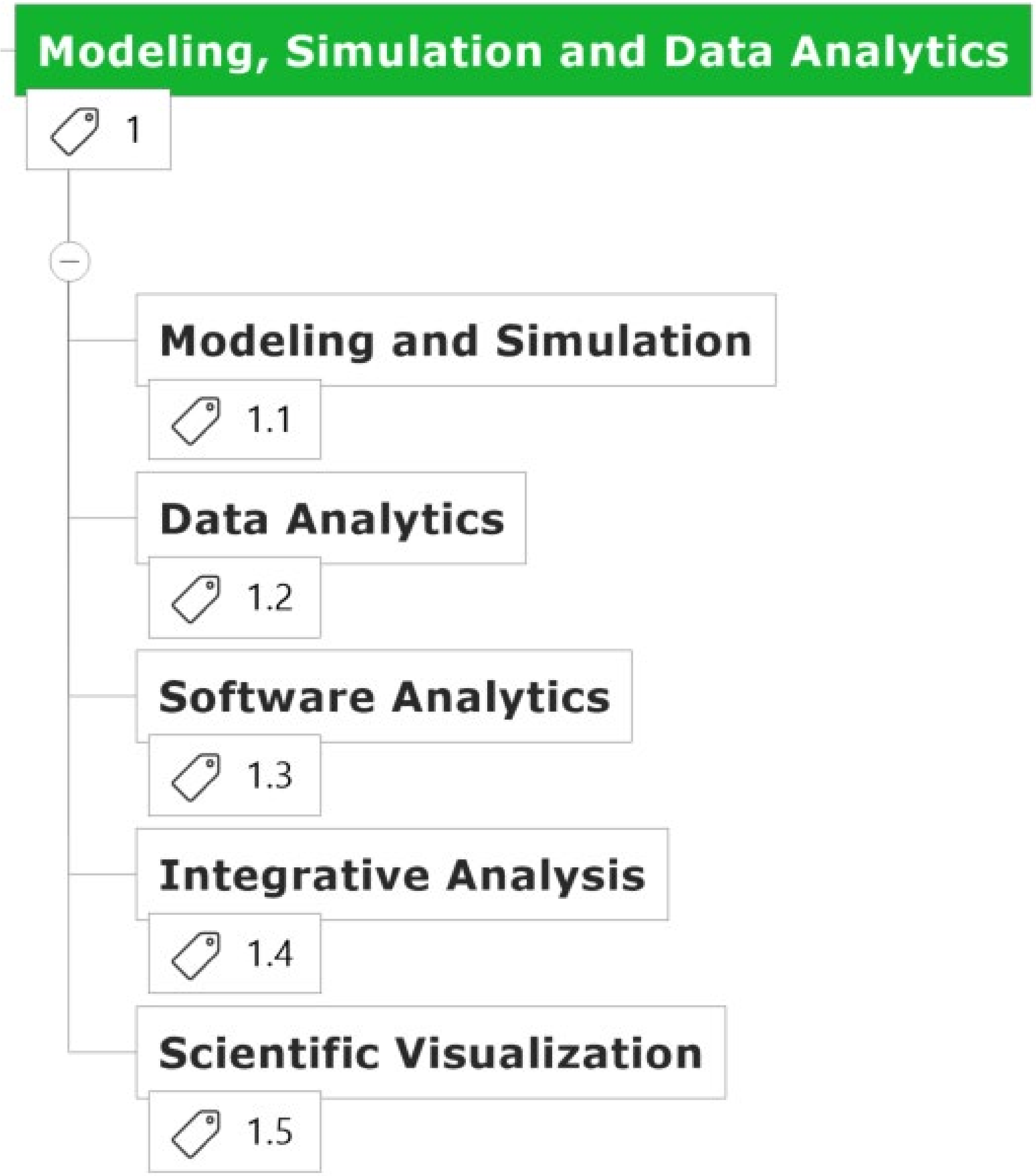
1. Modeling, Simulation and Data Analytics
2. Technology Research Software
3. Research Infrastructure Software

Let‘s take a look at the sub-categories via the mindmap.

Refinement of Category 1

Modeling, Simulation and Data Analytics of, e.g., physical, chemical, social, or biological processes.

- **1.1 Modeling and Simulation** (e.g., numerical modeling, agent-based modeling)
- **1.2 Data Analytics**, on observation and simulation data, with statistical analysis and machine learning as methods
- **1.3 Software Analytics** (static, dynamic, evolution, repository mining)
- **1.4 Integrative Analysis** (data assimilation, decision analysis)
- **1.5 Scientific Visualization**



Related:
Defining the roles of research software
[van Nieuwpoort 2022, van Nieuwpoort and Katz 2023]

Research software is a component of our instruments	Category 3.1
Research software *is* the instrument	Category 1 & 3
Research software analyses research data	Category 1.2
Research software presents research results	Category 1.5
Research software assembles or integrates existing components into a working whole	Category 3.3
Research software is infrastructure or an underlying tool	Category 3
Research software facilitates distinctively research-oriented collaboration	Category 3.6 – 3.8

Category 2 not included.

Update: [van Nieuwpoort and Katz 2024]

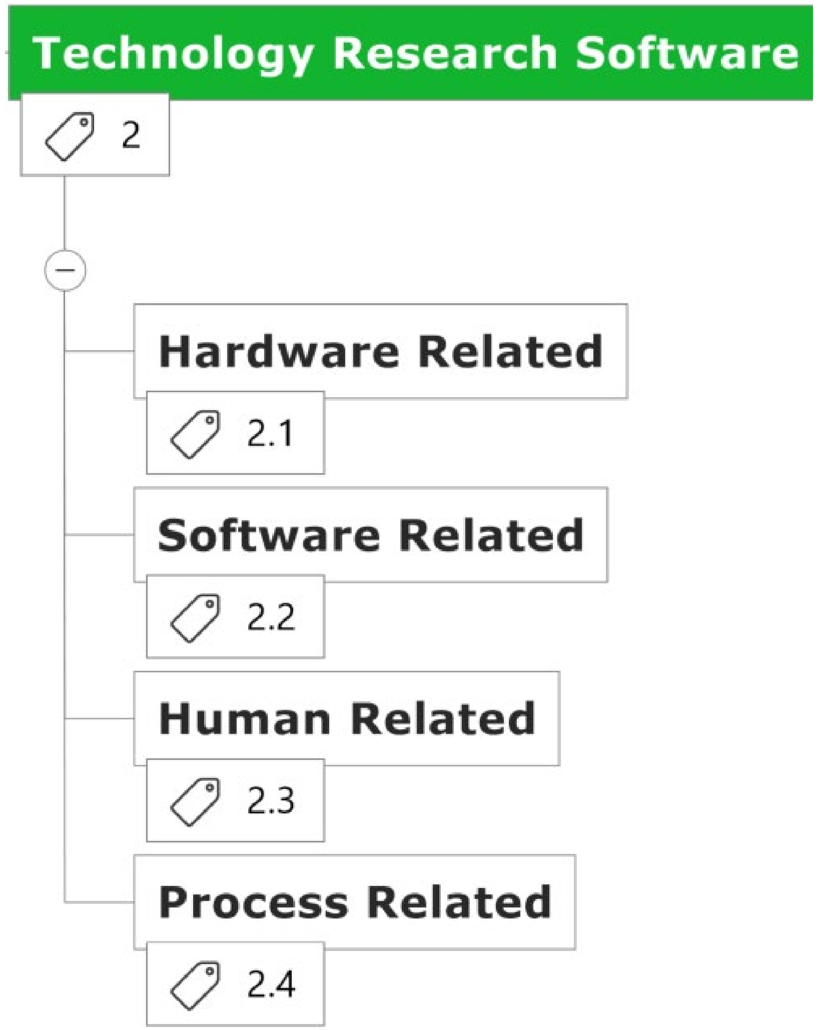
Research software is a component of our instruments
Research software *is* the instrument
Research software analyses research data
Research software presents research results
Research software assembles or integrates existing components into a working whole
Research software is infrastructure or an underlying tool
Research software facilitates distinctively research-oriented collaboration
Research software *itself* is a research tool for technology research

- In **technology research** (most often in computer science, and also in other disciplines), research software often plays a special role.
- Here, the research software **itself** is a key research tool
- For example, it can be a software **prototype** that **demonstrates** or explores a novel technological concept.
- An example is a computer science researcher who is researching **compiler technology**, with the idea of examining the performance of different options in programming language design.
- In this case, the **prototype compiler** is research software, since it is an artifact produced by computer science research. We therefore call this class of software "technology research software".

Refinement of Category 2

Technology Research Software in science and engineering research may be related to target contexts:

- **2.1 Hardware Related** (usually as embedded software)
- **2.2 Software Related** (e.g., as part of an operating system)
- **2.3 Human Related** (with a user interface)
- **2.4 Process Related** (e.g., as part of a business, development or production processes)

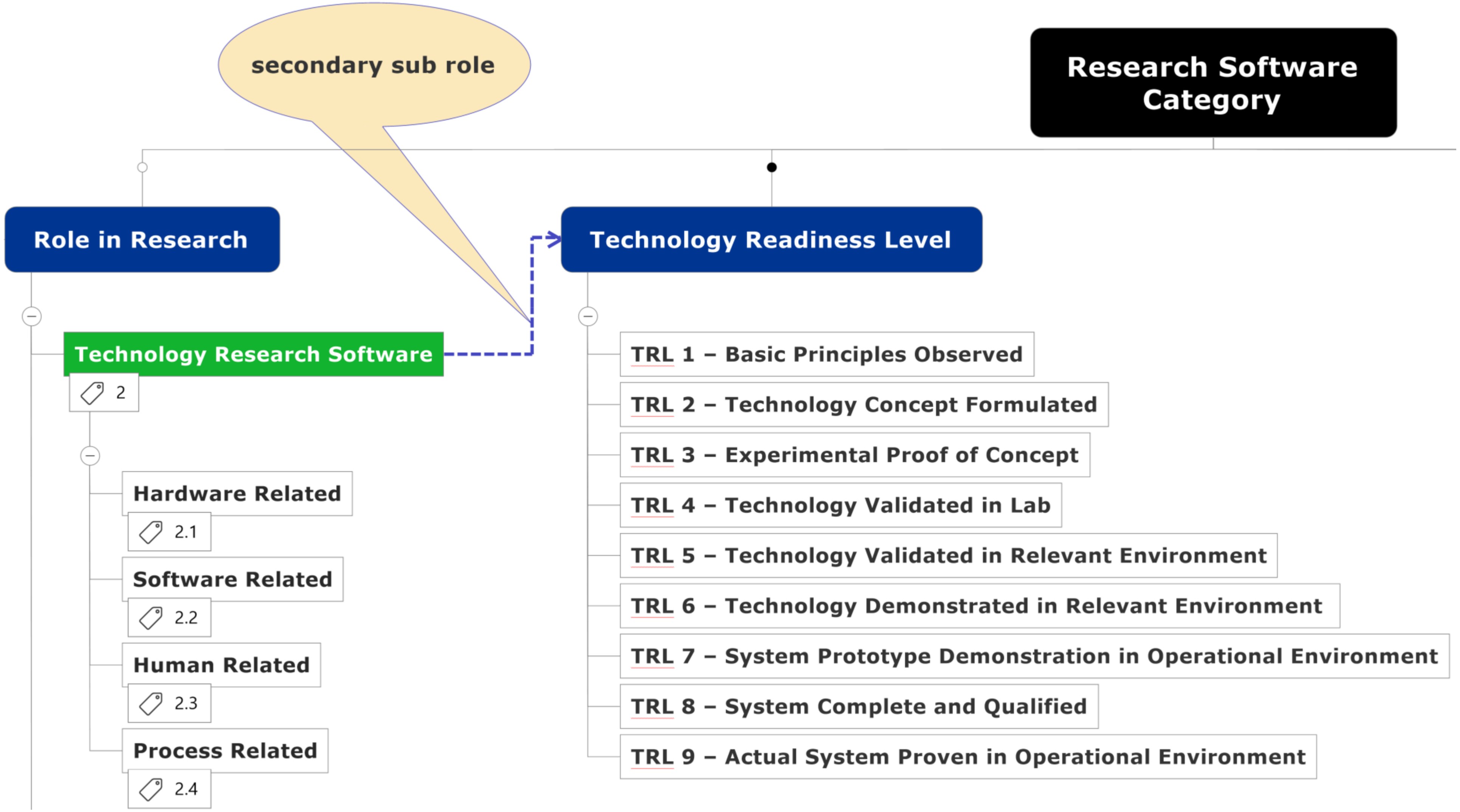


Category 2: Technology Research Software

- "**Technology** is the application of conceptual knowledge for achieving practical goals, especially in a reproducible way.
  - The word technology can also mean the products resulting from such efforts, including both tangible tools such as utensils or machines, and intangible ones such as **software**." https://en.wikipedia.org/wiki/Technology
- **Engineering Research** (AKA Design Science) is research that invents and evaluates **technological** artifacts.[1]
  - Could also be called Technology Research, see [van Nieuwpoort and Katz 2024].
- The refinement via "**Technology Readiness Levels**" should be appropriate.

[1] https://github.com/acmsigsoft/EmpiricalStandards/blob/master/docs/standards/EngineeringResearch.md

Technology Readiness Levels as Secondary Sub Roles



Technology Readiness Levels (TRL)

TRL 1 – basic principles observed
TRL 2 – technology concept formulated
TRL 3 – experimental proof of concept
TRL 4 – technology validated in lab
TRL 5 – technology validated in relevant environment
TRL 6 – technology demonstrated in relevant environment
TRL 7 – system prototype demonstration in operational environment
TRL 8 – system complete and qualified
TRL 9 – actual system proven in operational environment

} Technology Research Software

[Rose et al. 2017]

Technology Research Software Secondary Sub Roles

- The TRLs constitute sub roles of technology research software.
- One specific technology research software may take several such sub roles over its lifecyle, with increasing “readiness”.
- It may also take several roles at the same time, within different contexts:
  - In one project context, it may serve as “Experimental Proof of Concept” (TRL 3);
  - in another project, it may already serve as a “Technology Validated in Lab” (TRL 4).
  - Eventually, a technology research software may even become an “Actual System Proven in Operational Environment” (TRL 9).

“Modeling and Simulation Research Software” vs. “Technology Research Software”

The difference between the categories “Modeling and Simulation” and “Technology Research Software” (without consideration of the TRL sub roles) may be illustrated, for instance, with control engineering research:

- As a control engineering researcher, you may build a simulation of a control system.
- As a control engineering researcher, you may also build an actual control system as a new software system.
  - In an automation lab, this researcher may then experiment with this system (not with the simulation of the system).
  - If this system (which is a technology research software) matures, it may reach higher TRLs.

Here, both, the simulation and the actual control system are research software.

- The simulation software may even become part of the actual control system (for instance, for prediction), turning it into technology [research software].

Category 3:
Research Infrastructure Software

3.1 Control and Monitoring Software for complex experiments and instruments. This includes embedded control software, as well as native and web-based monitoring software

3.2 Data Collection and Generation (survey software, sensor-based data collection, synthetic data generation, etc.)

3.3 Pipelines and Tools

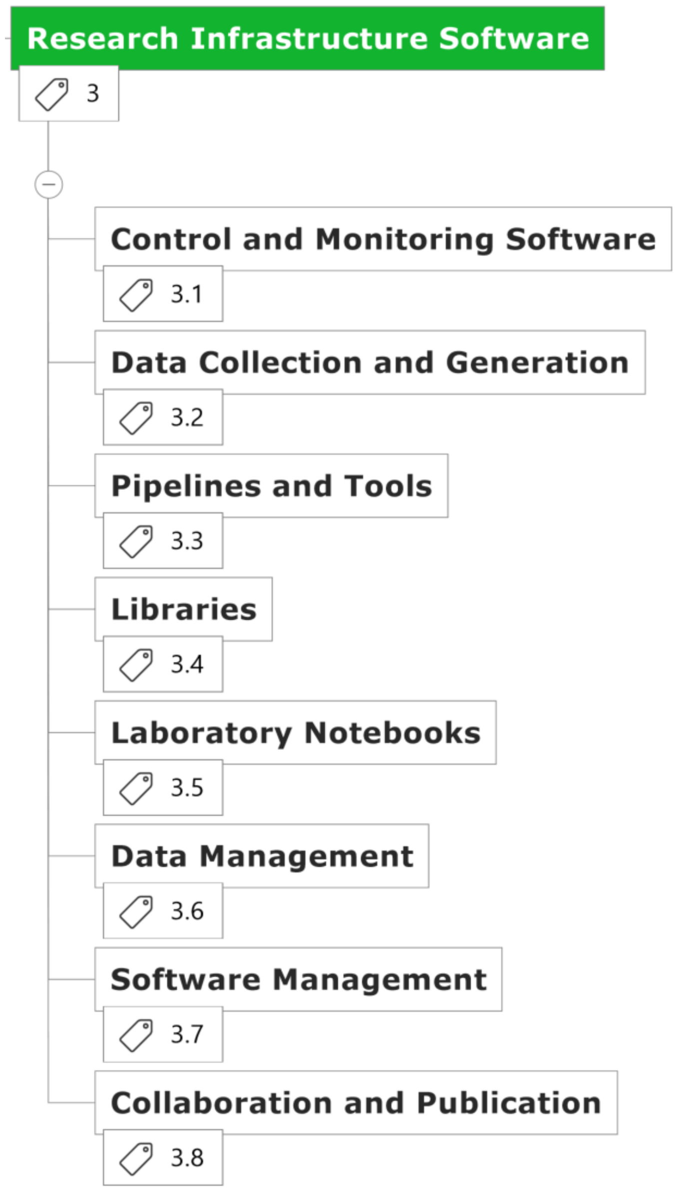
3.4 Libraries, for instance for high performance computing

3.5 Laboratory Notebooks

3.6 Data Management

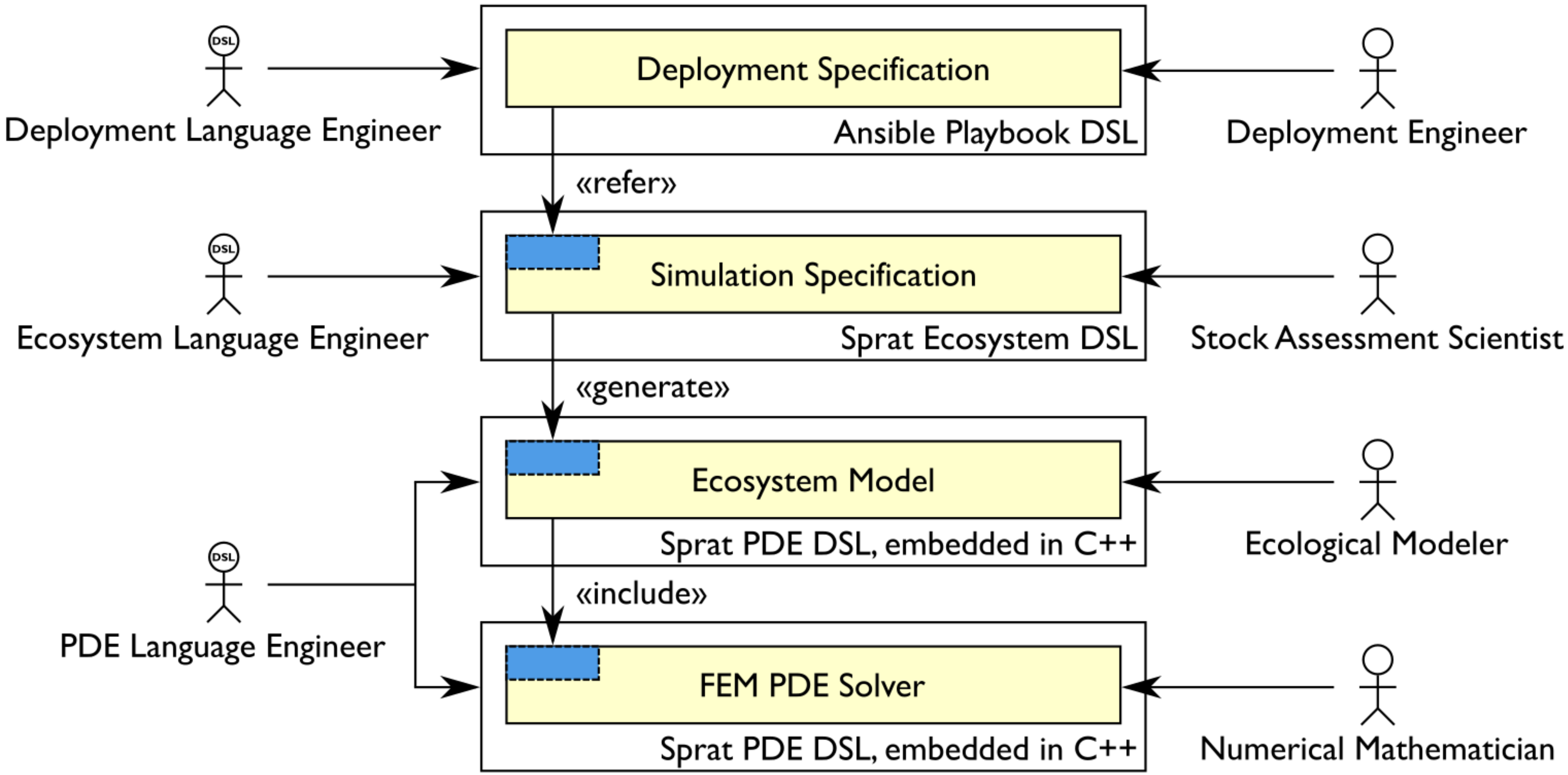
3.7 Software Management

3.8 Collaboration and Publication



Research Software Examples

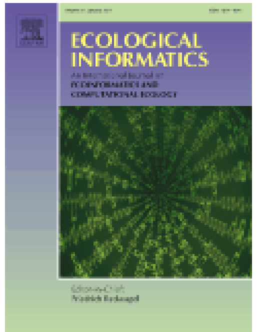
Example for Category 1.1 (Modeling and simulation): The Sprat Marine Ecosystem Modeling Languages

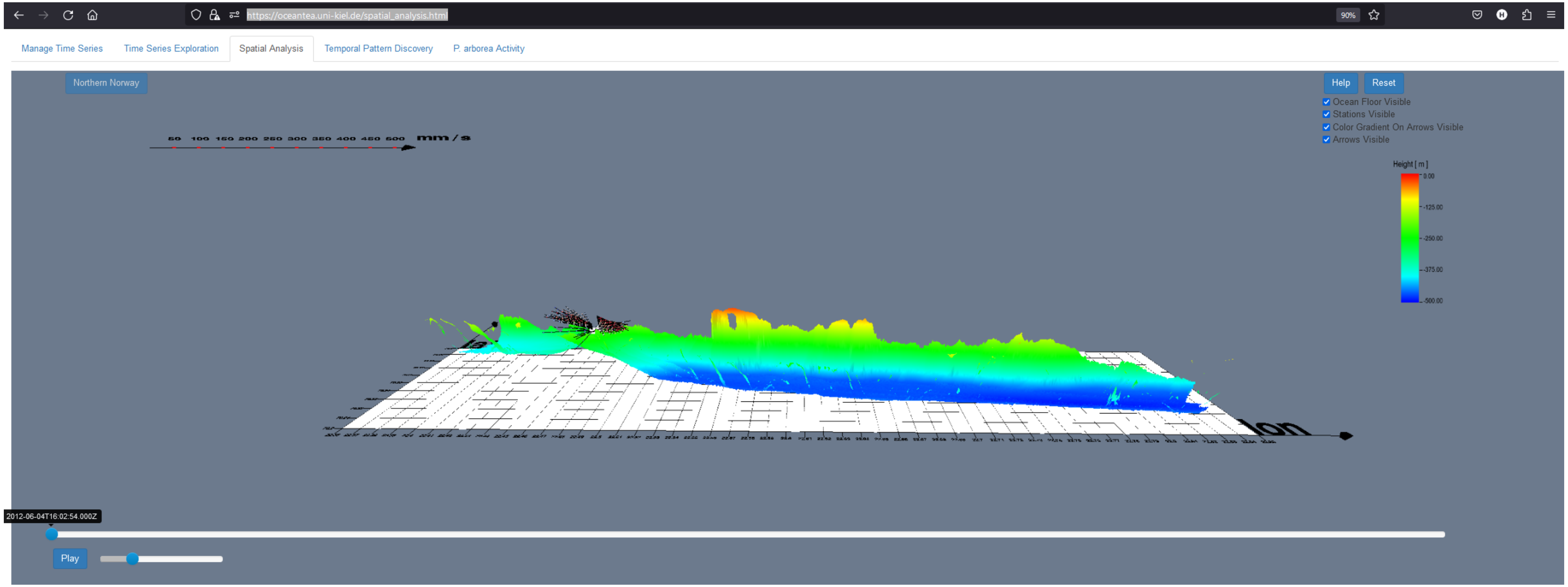


SPRAT

[Johanson & Hasselbring 2014a,b, 2016b, Johanson et al. 2017a, Johanson & Hasselbring 2017]

Example for Category 1.2 (Data Analytics): OceanTEA: Analyzing Ocean Observation Data





Paper on the analysis results: [Johanson et al. 2017b]

Paper on the software architecture: [Johanson et al. 2016a]

Code: https://github.com/cau-se/oceantea

future ocean
KIEL MARINE SCIENCES

Examples for Category 2 (Technology Research Software)

https://github.com/kieker-monitoring

Kieker: A monitoring framework for software engineering research
[van Hoorn et al. 2012, Hasselbring and van Hoorn 2020]

https://github.com/ExplorViz

ExplorViz: Research on software visualization, comprehension and collaboration
[Hasselbring et al. 2020c]

https://www.theodolite.rocks

The Theodolite Scalability Benchmarking Framework
[Henning and Hasselbring 2021, 2022, 2024]

Multi-dimensional categorization of the **Kieker** observability and monitoring framework:

Role	Readiness	Developer	Dissemination
1.3 Software Analytics 2.2 Software Related	TRL 4 [42], [43], [44], [45], [46], [47], [48], [49], [50], [51], [52], [53] TRL 5 [54] TRL 6 [55]	Community SustainKieker Gefördert durch DFG Deutsche Forschungsgemeinschaft	Open Source

Multi-dimensional categorization of the **ExplorViz** software visualization tool:

Role	Readiness	Developer	Dissemination
1.3 Software Analytics 1.5 Scientific Visualization 2.2 Software Related	TRL 4 [31], [32], [33], [34], [35], [36] TRL 5 [37]	Local Research Group	Open Source Software as a Service [38]

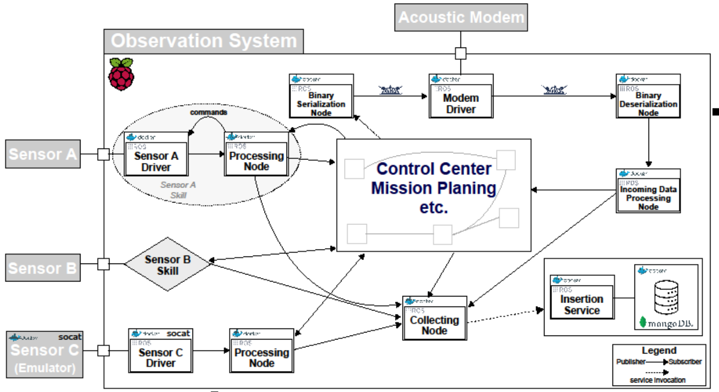
Multi-dimensional categorization of the **Theodolite** benchmarking framework:

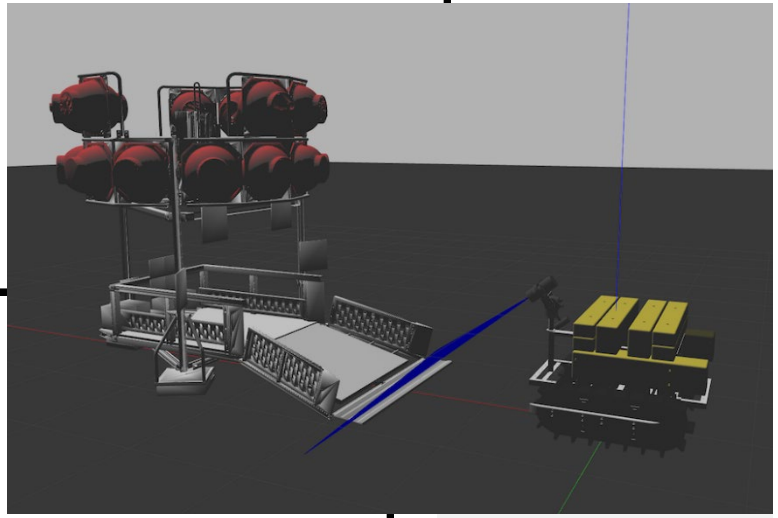
Role	Readiness	Developer	Dissemination
2.2 Software Related 3.3 Pipelines and Tools	TRL 4 [86] TRL 5 [87], [88]	Project Group	Open Source

[Hasselbring et al. 2024]

Example for Category 3.1 (Control & Monitoring): Software for Ocean Observation Robotics

Digital Twin Prototype





Digital Twin

Physical Twin

[Barbie et al. 2021]

GEOMAR

Multi-Dimensional Categorization of the ARCHES Digital Twin Framework

Role	Readiness	Developer	Dissemination
2.1 Hardware Related 3.1 Control and Monitoring Software	TRL 4 [25] TRL 5 [26] TRL 7 [24]	Project Group	Open Source

[Hasselbring et al. 2024]

https://irser.github.io/

RSECON24

Newcastle, Sept 03-05, ’24

Outlook: Research Software Engineering Research

Research Software Engineering

Software Engineering Research

Research Software Engineering Research aims at understanding and improving how software is developed for research.

RSE Research, in short [Felderer et al. 2023, 2025].

Sample RSE Research Question:
“**Which categories of research software require which software architecture structures?** ”

BETTER SOFTWARE ARCHITECTURE
BETTER SOFTWARE
BETTER RESEARCH
www.software.ac.uk

COMMUNICATIONS OF THE ACM
CACM.ACM.ORG
DOI: 10.1145/3685265

BY WILHELM HASSELBRING ET AL.

Investigating Research Software Engineering: Toward RSE Research

Research software engineering research aims at understanding and improving how software is developed for research.

[Felderer et al. 2025]

Slides

https://oceanrep.geomar.de/id/eprint/61416/

References

[Barbie et al. 2021] Barbie, A., Pech, N., Hasselbring, W., Flögel, S., Wenzhöfer, F., Walter, M., Shchekinova, E., Busse, M., Türk, M., Hofbauer, M. und Sommer, S.: “Developing an Underwater Network of Ocean Observation Systems with Digital Twin Prototypes - A Field Report from the Baltic Sea.” IEEE Internet Computing. 2021. DOI https://doi.org/10.1109/MIC.2021.3065245

[Chue Hong et al. 2022] N. P., Chue Hong, et al. (2022). FAIR Principles for Research Software version 1.0. (FAIR4RS Principles v1.0). Research Data Alliance. DOI https://doi.org/10.15497/RDA00068

[Felderer et al. 2023] Felderer, M., Goedicke, M., Grunske, L., Hasselbring, W., Lamprecht, A. L. und Rumpe, B.: “Toward Research Software Engineering Research”. 2023. DOI https://doi.org/10.5281/ZENODO.8020525.

[Felderer et al. 2025] Felderer, M., Goedicke, M., Grunske, L., Hasselbring, W., Lamprecht, A. L. und Rumpe, B.: “Investigating research software engineering: Toward RSE Research”. Communications of the ACM, 2025. DOI https://doi.org/10.1145/3685265

[Hasselbring et al. 2020a] W. Hasselbring, L. Carr, S. Hettrick, H. Packer, T. Tiropanis: “Open Source Research Software”. In: Computer, 53 (8), pp. 84-88. 2020. DOI https://doi.org/10.1109/MC.2020.2998235

[Hasselbring et al. 2020b] W. Hasselbring, L. Carr, S. Hettrick, H. Packer, T. Tiropanis: “From FAIR Research Data toward FAIR and Open Research Software”, it - Information Technology, 2020. DOI https://doi.org/10.1515/itit-2019-0040

[Hasselbring et al. 2020c] Hasselbring, W., Krause, A., Zirkelbach, C.: “ExplorViz: Research on software visualization, comprehension and collaboration.” In: Software Impacts, 6, 2020. DOI https://doi.org/10.1016/j.simpa.2020.100034.

[Hasselbring et al. 2024] Hasselbring, W., Druskat, S., Bernoth, J., Betker, P., Felderer, M., Ferenz, S., Hermann, B., Lamprecht, A. L., Linxweiler, J., Prat, A., Rumpe, B., Schoening-Stierand, K., Yang, S.: “Multi-dimensional categorization of research software with examples,” Zenodo, 2024. DOI: https://doi.org/10.5281/zenodo.14082554

[Hasselbring and van Hoorn 2020] Hasselbring, W., van Hoorn, A.: “Kieker: A monitoring framework for software engineering research.” In: Software Impacts, 5, 2020. pp. 1-5. DOI https://doi.org/10.1016/j.simpa.2020.100019

[Henning and Hasselbring 2021] Henning, S., Hasselbring, W.: “Theodolite: Scalability Benchmarking of Distributed Stream Processing Engines in Microservice Architectures.” In: Big Data Research, 25 (100209), 2021. pp. 1-17. DOI https://doi.org/10.1016/j.bdr.2021.100209

[Henning and Hasselbring 2022] Henning, S. und Hasselbring, W.: “A configurable method for benchmarking scalability of cloud-native applications.” In: Empirical Software Engineering, 27 (6), 2022. p. 143. DOI https://doi.org/10.1007/s10664-022-10162-1

References

[Henning and Hasselbring 2024] Henning, S., Hasselbring, W.: “Benchmarking scalability of stream processing frameworks deployed as microservices in the cloud.” In: Journal of Systems and Software, 208(111879), 2024. DOI https://doi.org/10.1016/j.jss.2023.111879.

[Johanson & Hasselbring 2014a] A. Johanson, W. Hasselbring: “Hierarchical Combination of Internal and External Domain-Specific Languages for Scientific Computing”. In: International Workshop on DSL Architecting & DSL-Based Architectures (DADA'14), August 2014, Vienna, Austria, pp. 17:1-17:8. DOI https://doi.org/10.1145/2642803.2642820

[Johanson & Hasselbring 2014b] A. Johanson, W. Hasselbring: “Sprat: Hierarchies of Domain-Specific Languages for Marine Ecosystem Simulation Engineering”. In: Spring Simulation Multi-Conference (SpringSim 2014), April 2014, Tampa, Florida, USA, pp. 187-192.

[Johanson et al. 2016a] A. Johanson, S. Flögel, C. Dullo, W. Hasselbring: “OceanTEA: Exploring Ocean-Derived Climate Data Using Microservices”. In: Sixth International Workshop on Climate Informatics (CI 2016), 2016, DOI http://dx.doi.org/10.5065/D6K072N6

[Johanson et al. 2016b] A. Johanson, W. Hasselbring, A. Oschlies, B. Worm: “Evaluating Hierarchical Domain-Specific Languages for Computational Science: Applying the Sprat Approach to a Marine Ecosystem Model”. In: Software Engineering for Science. CRC Press. 175-200.

[Johanson et al. 2017a] A. Johanson, A. Oschlies, W. Hasselbring, A. Worm: “SPRAT: A spatially-explicit marine ecosystem model based on population balance equations”, In: Ecological Modelling, 349, pp. 11-25, 2017. DOI https://doi.org/10.1016/j.ecolmodel.2017.01.020

[Johanson et al. 2017b] A. Johanson, S. Flögel, C. Dullo, P. Linke, W. Hasselbring: “Modeling Polyp Activity of Paragorgia arborea Using Supervised Learning”, In: Ecological Informatics, 39. pp. 109-118. 2017, DOI https://doi.org/10.1016/j.ecoinf.2017.02.007

[Johanson & Hasselbring 2017] A. Johanson, W. Hasselbring: “Effectiveness and efficiency of a domain-specific language for high-performance marine ecosystem simulation: a controlled experiment”, In: Empirical Software Engineering 22 (8). pp. 2206-2236, 2017. DOI https://doi.org/10.1007/s10664-016-9483-z

[Lamprecht et al. 2020] A.-L. Lamprecht et al.: “Towards FAIR principles for research software.” In: Data Science 3, 1 (June 2020), 37–59. DOI https://doi.org/10.3233/ds-190026

[Rose et al. 2017] A. D. Rose, M. Buna, C. Strazza, N. Olivieri, T. Stevens, L. Peeters, and D. Tawil-Jamault: “Technology readiness level: guidance principles for renewable energy technologies,” European Commission, Directorate General for Research and Innovation, 2017. DOI https://doi.org/10.2777/577767

References

[van Hoorn et al. 2012] A. van Hoorn, J. Waller, W. Hasselbring: “Kieker: A Framework for Application Performance Monitoring and Dynamic Software Analysis”. In: 3rd joint ACM/SPEC International Conference on Performance Engineering (ICPE'12). , April 22-25, 2012, Boston, Massachusetts, USA, pp. 247-248. DOI https://doi.org/10.1145/2188286.2188326.

[van Nieuwpoort 2022] R. van Nieuwpoort: “What does Research Software look like?”, Zenodo, 2022. DOI https://doi.org/10.5281/zenodo.7347700

[van Nieuwpoort and Katz 2023] R. van Nieuwpoort and D.S. Katz: “Defining the roles of research software”, Upstream, 2023. DOI https://doi.org/10.54900/9akm9y5-5ject5y

[van Nieuwpoort and Katz 2024] R. van Nieuwpoort and D.S. Katz: “Defining the roles of research software (Version 2)”, Upstream, 2024. DOI https://doi.org/10.54900/xdh2x-kj281