Research Software Categorization: Update

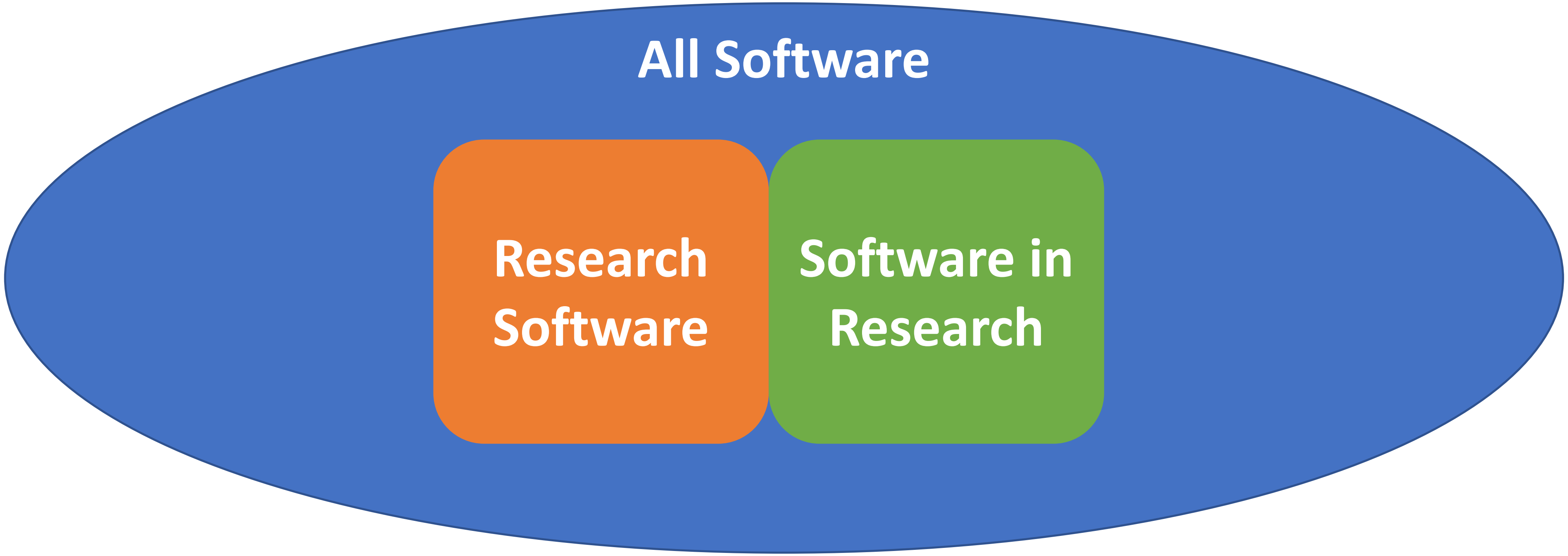
Wilhelm (Willi) Hasselbring

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/professor-wilhelm-hasselbring

Köln, October 15, 2024

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

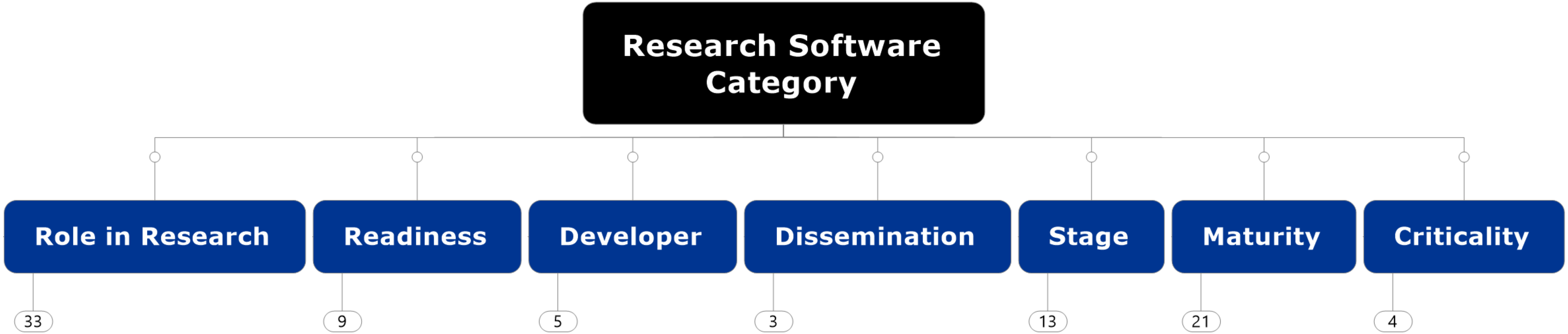
Purpose of Research Software Categories

Categories for research software may serve

- as a basis of **institutional guidelines** and checklists for research software development;
    - to better understand the different types of research software and their specific **quality** requirements;
    - to recommend appropriate software engineering **methods** for the individual categories;
- to design appropriate **teaching / education** programs for the individual categories;
- for a better assessment of existing software when deciding to **reuse** it;
- for research funding agencies, to define appropriate **funding** schemes;
- to define appropriate **metadata** labels for FAIR research software;
- in RSE Research, to provide a framework for classifying **research software artifacts**.

This list is not exhaustive.

Multi-Dimensional Categorization



[Hasselbring et al. 2024]

Roles of Research Software

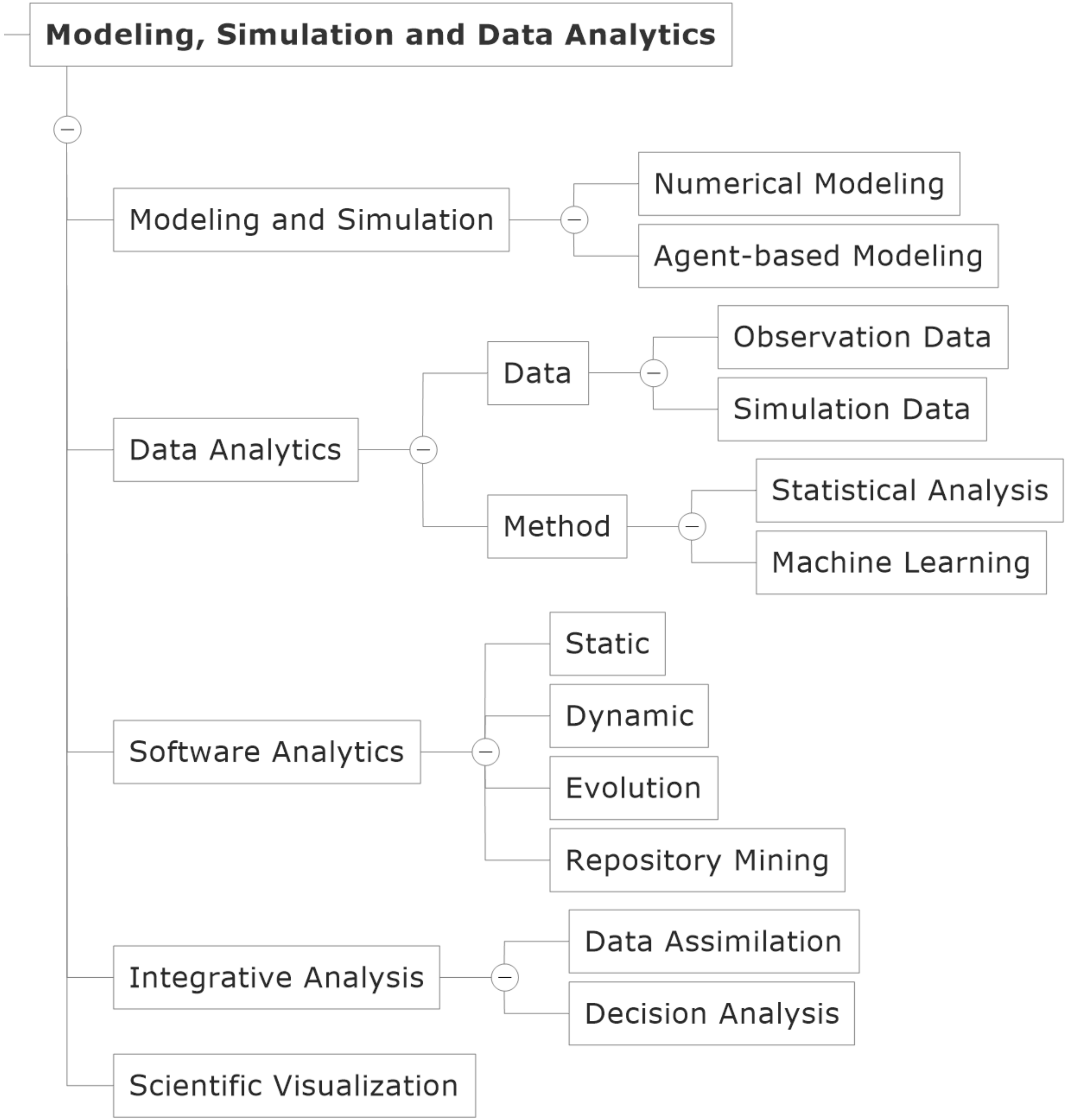
Research software’s roles mainly fall into one of the following top-level role categories (and sometimes combinations):

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

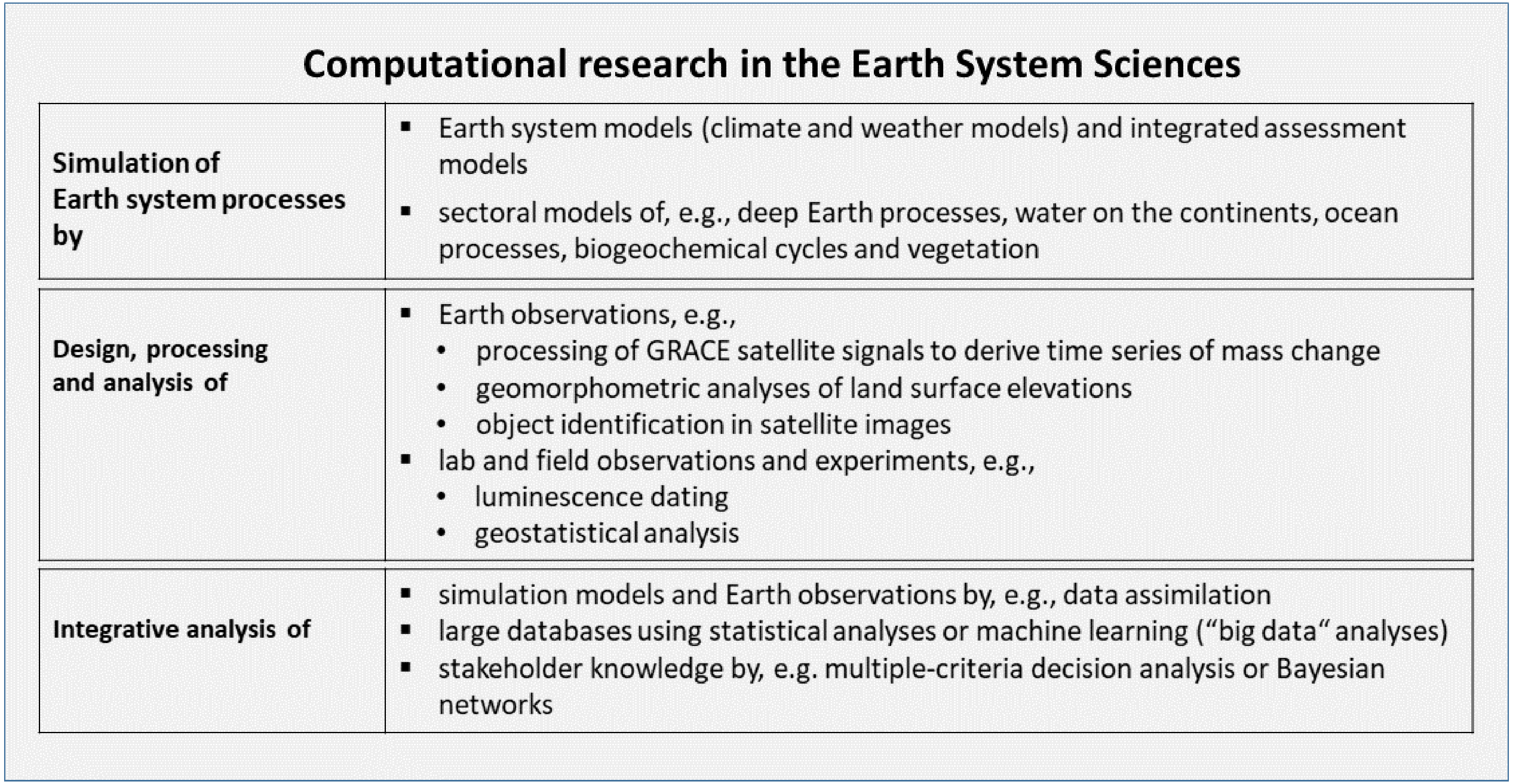
Let‘s take a look at the sub-categories via the mindmap.

Refinement of Category 1

1. **Modeling, Simulation and Data Analytics** of, e.g., physical, chemical, social, or biological processes in spatio-temporal contexts.
   1. Numerical and agent-based modeling and simulation (in silico experiments)
   2. Data Analytics (observation / simulation data, statistical analysis and machine learning)
   3. Software Analytics (static, dynamic, evolution, repository mining)
   4. Integrative Analysis (data assimilation, decision analysis)
   5. Scientific Visualization



Related:
Category 1 in Earth System Sciences



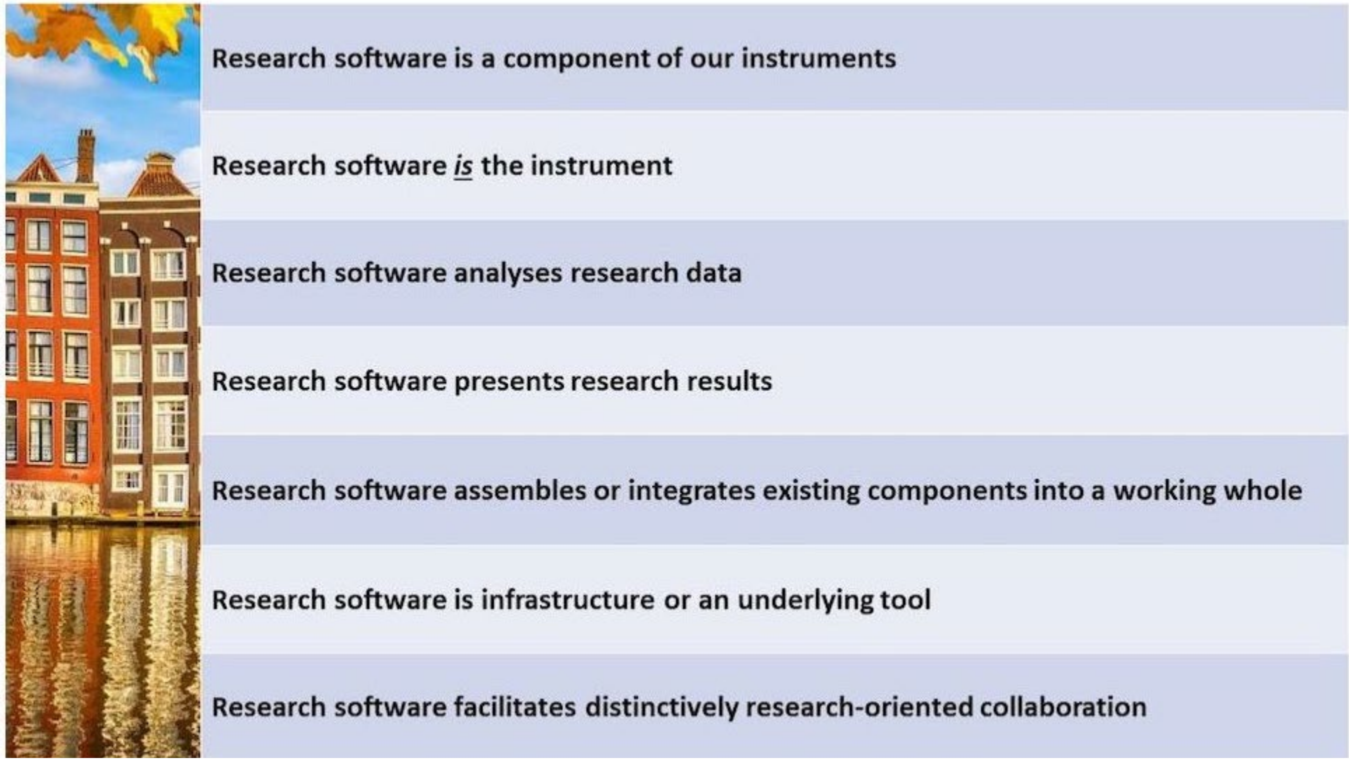
Computational research in the Earth System Sciences

Simulation of Earth system processes by	▪ Earth system models (climate and weather models) and integrated assessment models ▪ sectoral models of, e.g., deep Earth processes, water on the continents, ocean processes, biogeochemical cycles and vegetation
Design, processing and analysis of	▪ Earth observations, e.g., • processing of GRACE satellite signals to derive time series of mass change • geomorphometric analyses of land surface elevations • object identification in satellite images ▪ lab and field observations and experiments, e.g., • luminescence dating • geostatistical analysis
Integrative analysis of	▪ simulation models and Earth observations by, e.g., data assimilation ▪ large databases using statistical analyses or machine learning (“big data“ analyses) ▪ stakeholder knowledge by, e.g. multiple-criteria decision analysis or Bayesian networks

[Döll et al. 2023]

Related:
Defining the roles of research software

[van Nieuwpoort 2022, van Nieuwpoort and Katz 2023]

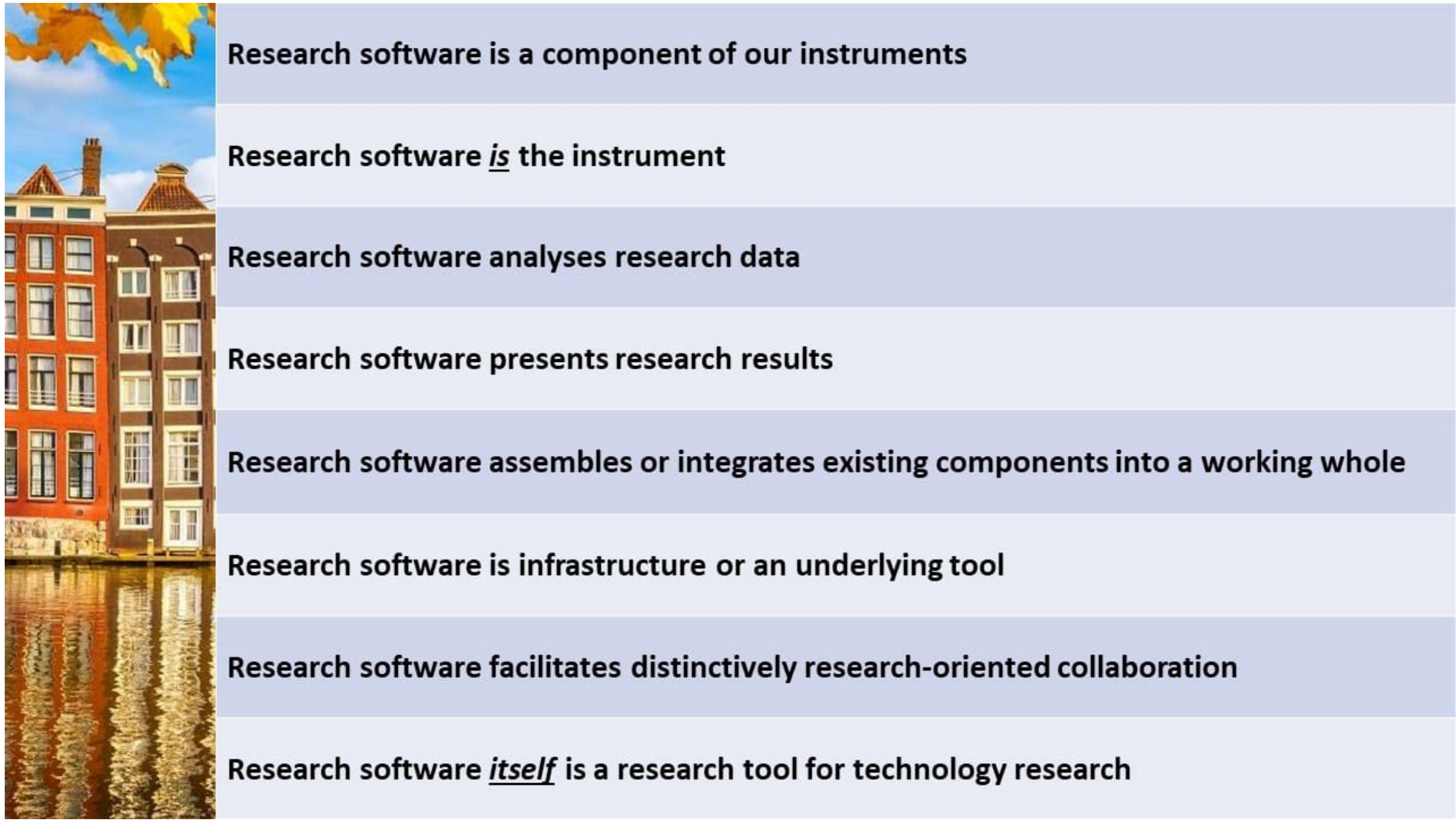


Research software is a component of our instruments	Category 3.1
Research software *is* the instrument	Category 1 & 3
Research software analyses research data	Category 1.2
Research software presents research results	Category 1.5
Research software assembles or integrates existing components into a working 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.

(meanwhile updated at https://doi.org/10.54900/xdh2x-kj281)

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

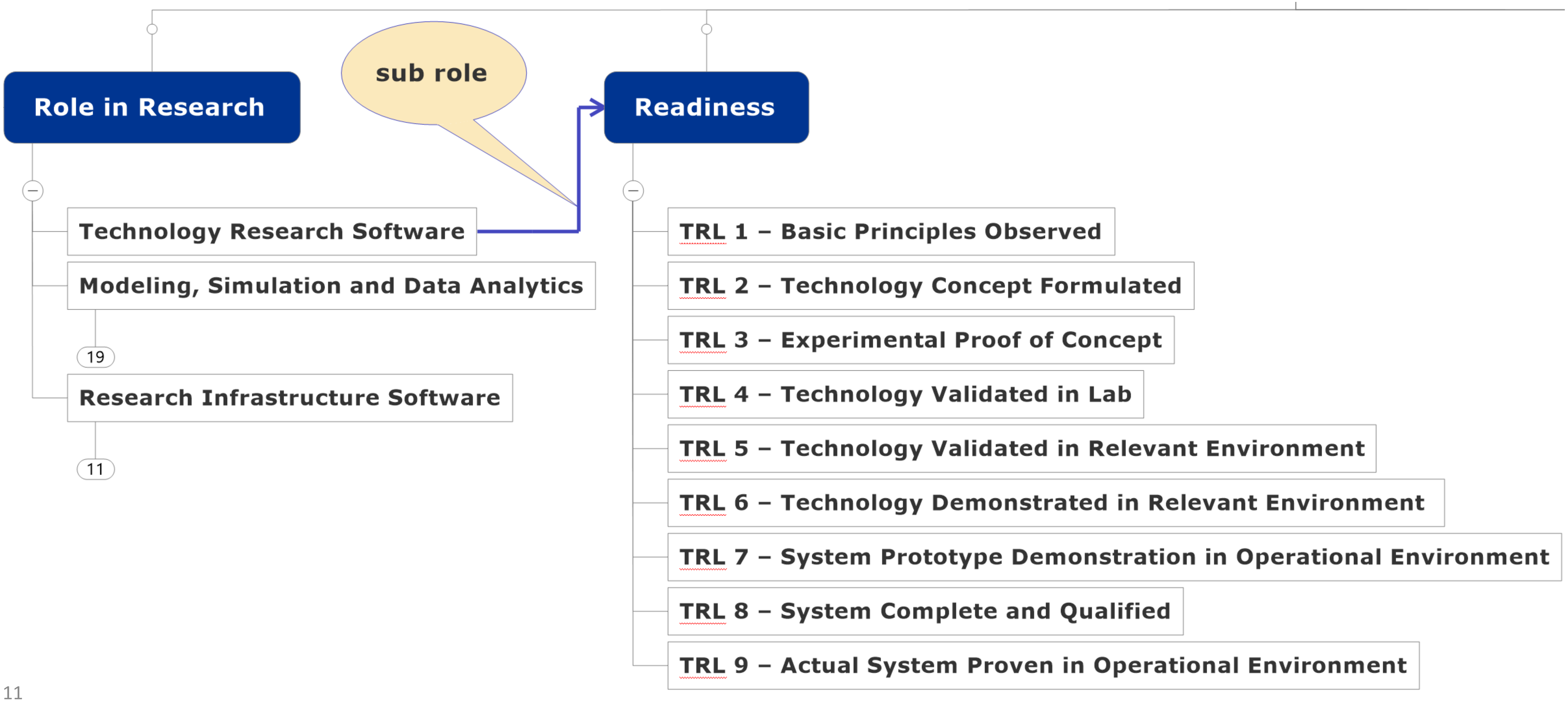
- In technology research (most often in computer science, and also in other disciplines), research software often plays a special role.
- Here, the research software itself is a key research tool
- For example, it can be a software prototype that demonstrates or explores a novel technological concept.
- 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”.

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]
- The refinement via “**Technology** Readiness Levels” should be appropriate.

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

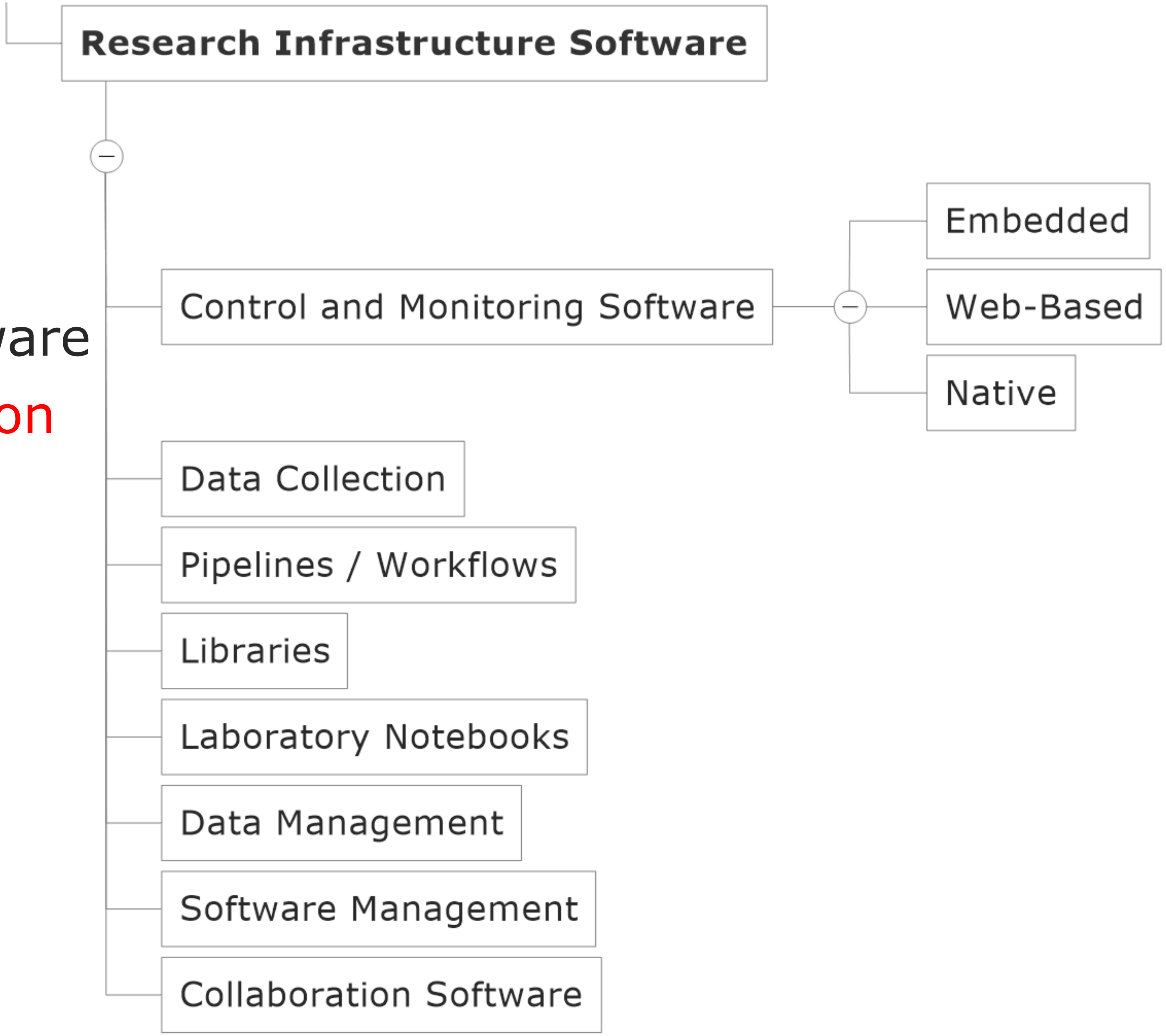
Technology Readiness Levels as Sub Roles



Role in Research
sub role
Readiness
Technology Research Software
Modeling, Simulation and Data Analytics
19
Research Infrastructure Software
11
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

Category 3

3.1. Control and Monitoring Software
3.2. Data Collection and Generation
3.3. Pipelines / Workflows
3.4. Libraries
3.5. Laboratory Notebooks
3.6. Data Management
3.7. Software Management
3.8. Collaboration and Publication



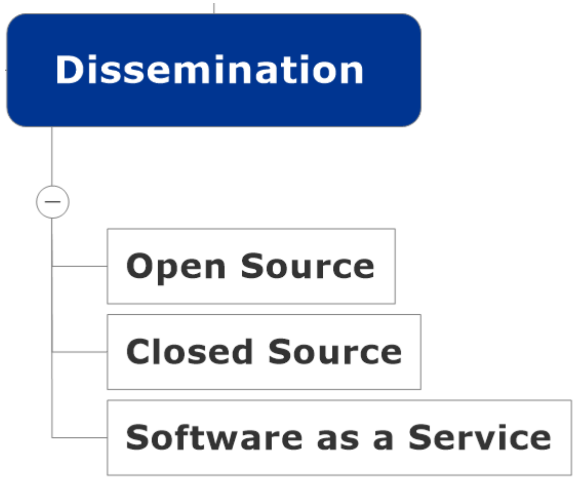
Developer & Dissemination Dimension

Individual Researcher

Local Research Group
Project Group

Community

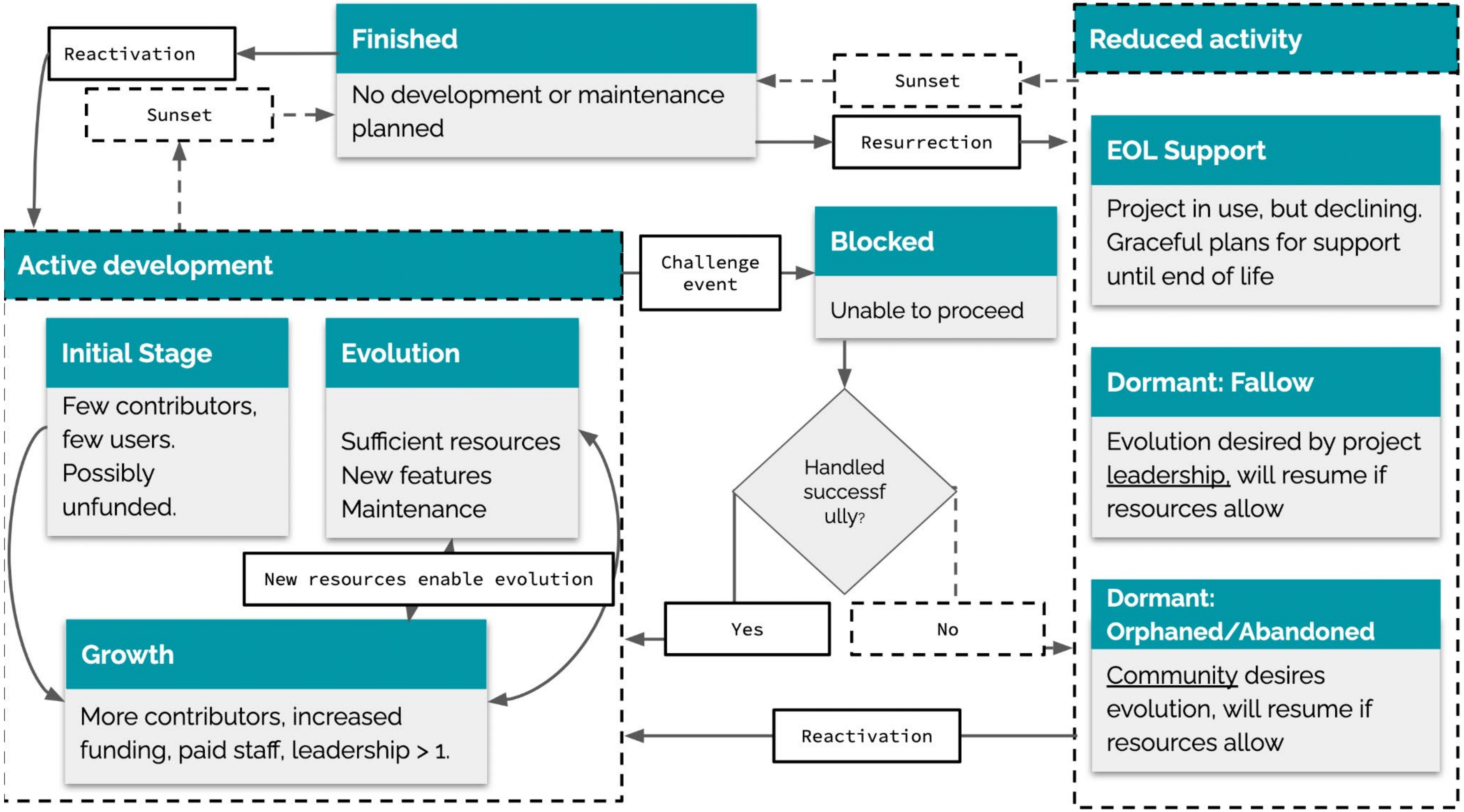
Contractor



Dissemination

- Open Source
- Closed Source
- Software as a Service

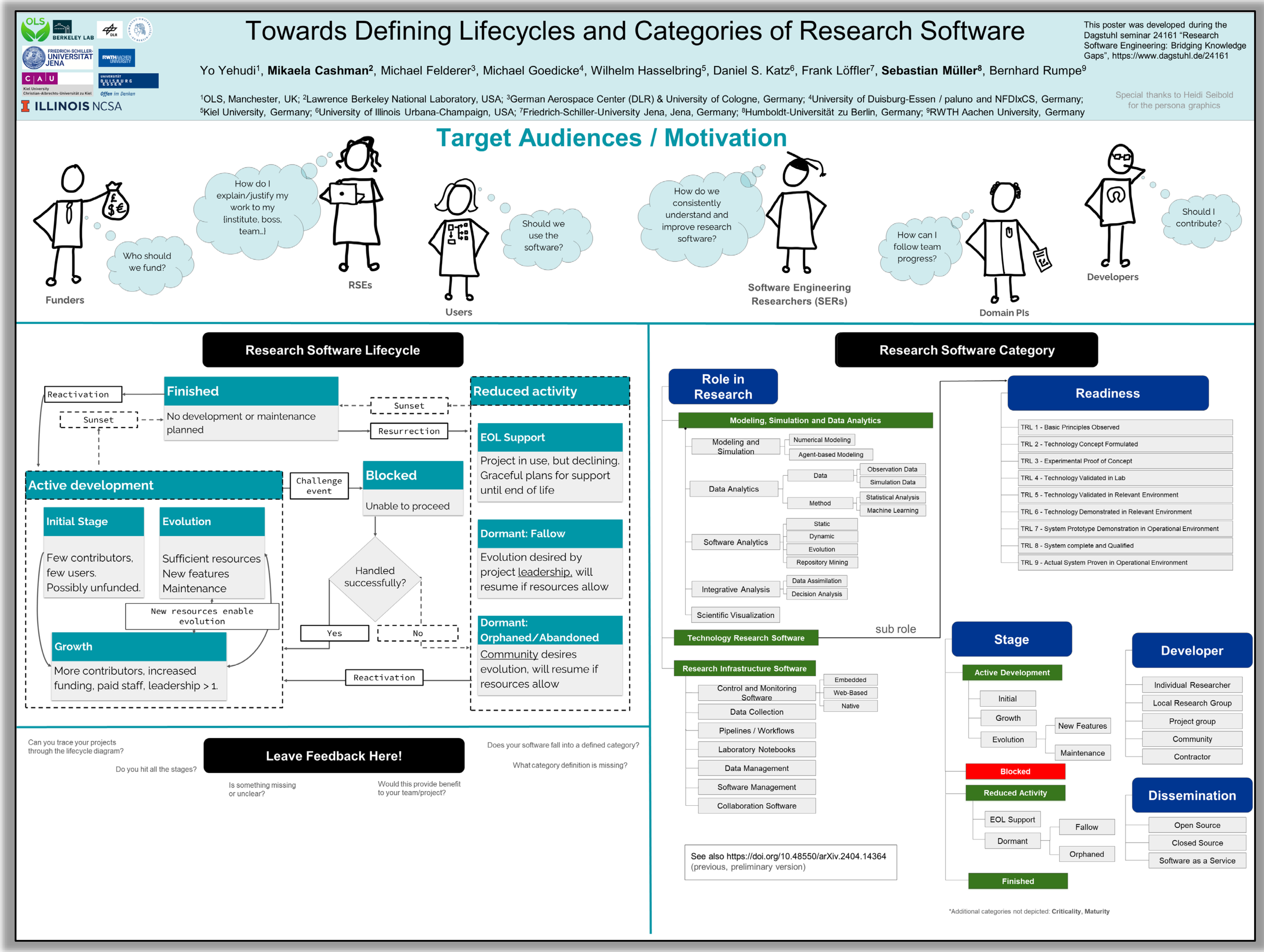
Stage Dimension:
Research Software Lifecycles


Reactivation
Finished
No development or maintenance planned
Sunset
Sunset
Resurrection
Reduced activity
EOL Support
Project in use, but declining. Graceful plans for support until end of life
Active development
Challenge event
Blocked
Unable to proceed
Initial Stage
Few contributors, few users. Possibly unfunded.
Evolution
Sufficient resources
New features
Maintenance
Handled successfully?
Dormant: Fallow
Evolution desired by project leadership, will resume if resources allow
New resources enable evolution
Yes
No
Growth
More contributors, increased funding, paid staff, leadership > 1.
Dormant: Orphaned/Abandoned
Community desires evolution, will resume if resources allow
Reactivation

Work in Progress with:
Yo Yehudi
Michael Goedicke
Sebastian Müller
Wilhelm Hasselbring
Bernhard Rumpe
Jan Linxweiler
Michael Felderer
Mikaela Cashman McDevitt
Daniel Katz
Frank Löffler

(Poster at US-RSE´24)

Poster



Towards Defining Lifecycles and Categories of Research Software

Yo Yehudi[1], **Mikaela Cashman[2]**, Michael Felderer[3], Michael Goedicke[4], Wilhelm Hasselbring[5], Daniel S. Katz[6], Frank Löffler[7], **Sebastian Müller[8]**, Bernhard Rumpe[9]

[1]OLS, Manchester, UK; [2]Lawrence Berkeley National Laboratory, USA; [3]German Aerospace Center (DLR) & University of Cologne, Germany; [4]University of Duisburg-Essen / paluno and NFDIxCS, Germany; [5]Kiel University, Germany; [6]University of Illinois Urbana-Champaign, USA; [7]Friedrich-Schiller-University Jena, Jena, Germany; [8]Humboldt-Universität zu Berlin, Germany; [9]RWTH Aachen University, Germany

This poster was developed during the Dagstuhl seminar 24161 “Research Software Engineering: Bridging Knowledge Gaps”, https://www.dagstuhl.de/24161

Special thanks to Heidi Seibold for the persona graphics

Target Audiences / Motivation

Funders: Who should we fund?

RSEs: How do I explain/justify my work to my [institute, boss, team..]

Users: Should we use the software?

Software Engineering Researchers (SERs): How do we consistently understand and improve research software?

Domain PIs: How can I follow team progress?

Developers: Should I contribute?

Research Software Lifecycle

Finished
No development or maintenance planned

Active development

Initial Stage
Few contributors, few users. Possibly unfunded.

Evolution
Sufficient resources
New features
Maintenance

Growth
More contributors, increased funding, paid staff, leadership > 1.

Blocked
Unable to proceed

Handled successfully?

Yes

No

Reduced activity

EOL Support
Project in use, but declining. Graceful plans for support until end of life

Dormant: Fallow
Evolution desired by project leadership, will resume if resources allow

Dormant: Orphaned/Abandoned
Community desires evolution, will resume if resources allow

Can you trace your projects through the lifecycle diagram?

Do you hit all the stages?

Leave Feedback Here!

Is something missing or unclear?

Would this provide benefit to your team/project?

Does your software fall into a defined category?

What category definition is missing?

Research Software Category

Role in Research

- **Modeling, Simulation and Data Analytics**
  - Modeling and Simulation
    - Numerical Modeling
    - Agent-based Modeling
  - Data Analytics
    - Data
      - Observation Data
      - Simulation Data
    - Method
      - Statistical Analysis
      - Machine Learning
  - Software Analytics
    - Static
    - Dynamic
    - Evolution
    - Repository Mining
  - Integrative Analysis
    - Data Assimilation
    - Decision Analysis
  - Scientific Visualization
- **Technology Research Software**
- **Research Infrastructure Software**
  - Control and Monitoring Software
    - Embedded
    - Web-Based
    - Native
  - Data Collection
  - Pipelines / Workflows
  - Laboratory Notebooks
  - Data Management
  - Software Management
  - Collaboration Software

sub role

Readiness

- 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

Stage

- **Active Development**
  - Initial
  - Growth
  - Evolution
    - New Features
    - Maintenance
- **Blocked**
- **Reduced Activity**
  - EOL Support
  - Dormant
    - Fallow
    - Orphaned
- **Finished**

Developer

- Individual Researcher
- Local Research Group
- Project group
- Community
- Contractor

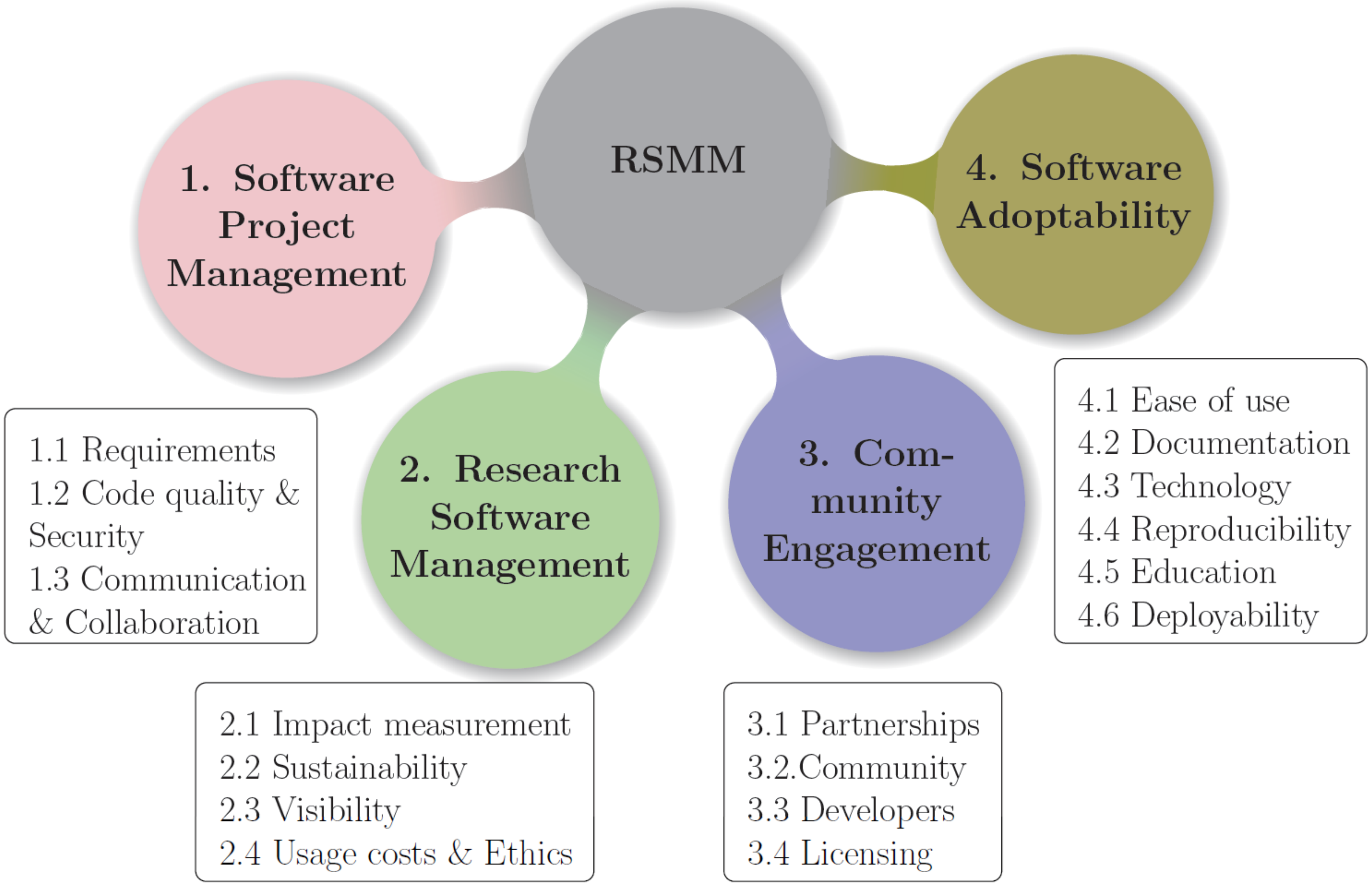
Dissemination

- Open Source
- Closed Source
- Software as a Service

See also https://doi.org/10.48550/arXiv.2404.14364 (previous, preliminary version)

*Additional categories not depicted: **Criticality, Maturity**

Maturity: Assessment via RSMM



RSMM

1. Software Project Management

1.1 Requirements
1.2 Code quality & Security
1.3 Communication & Collaboration

2. Research Software Management

2.1 Impact measurement
2.2 Sustainability
2.3 Visibility
2.4 Usage costs & Ethics

3. Community Engagement

3.1 Partnerships
3.2.Community
3.3 Developers
3.4 Licensing

4. Software Adoptability

4.1 Ease of use
4.2 Documentation
4.3 Technology
4.4 Reproducibility
4.5 Education
4.6 Deployability

4 focus areas and 17 capabilities: [Deekshitha et al. 2024]

(Framework for research software assessment based on COBIT: [zu Castell et al. 2024])

Role / Readiness / Maturity: 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

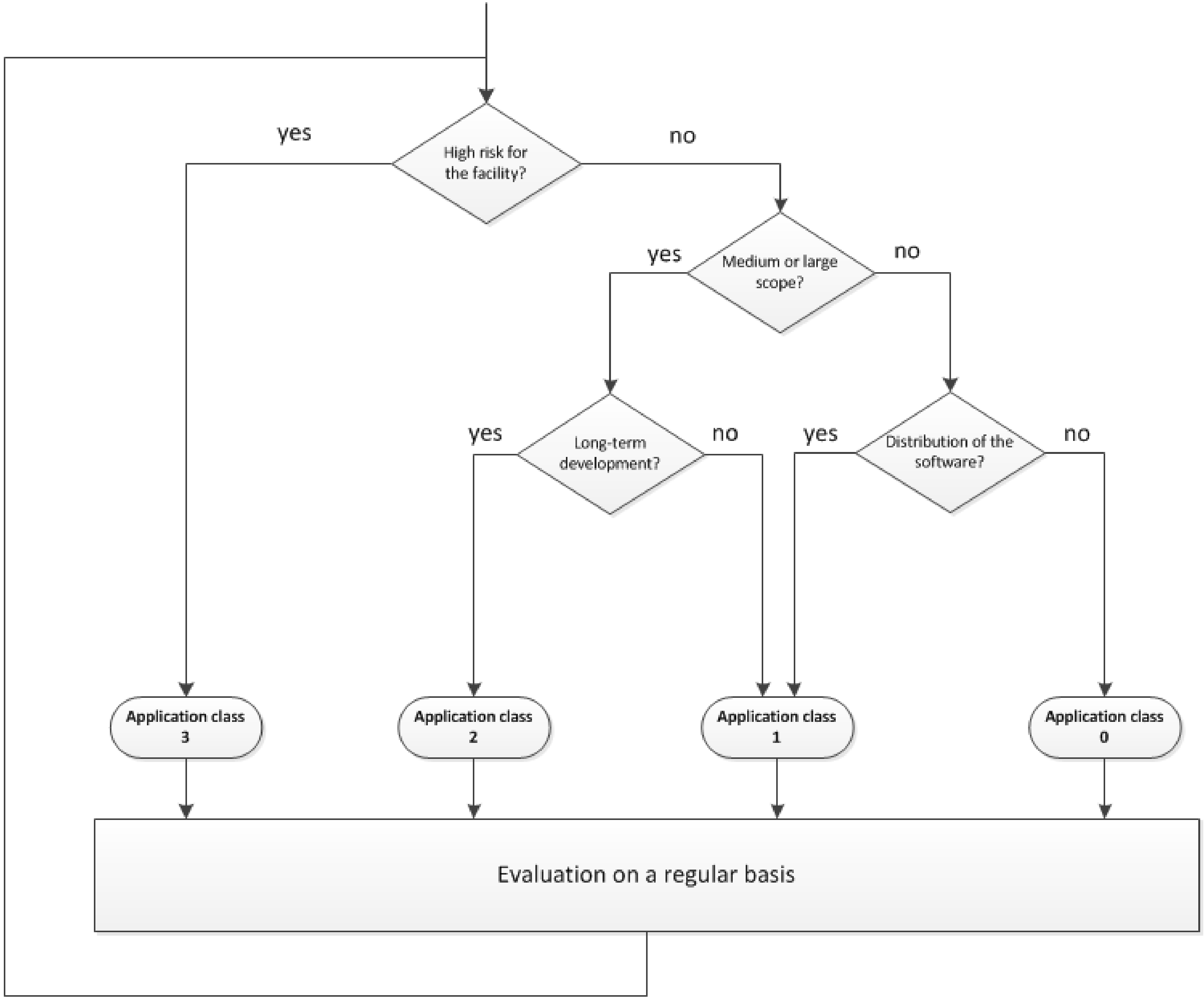
[Rose et al. 2017]

Related: Application Classes in Institutional Guidelines

Application Class 0	Small scope, personal use	• Scripts to process data for a publication. • Simple administrative scripts to automate specific tasks • Software that demonstrates or tests certain functions
Application Class 1	Narrow scope, beyond personal use	• Software from Bachelor/Master/PhD theses • Software from smaller/shorter research projects
Application Class 2	Extended scope, wider use	• Software from longer-term research projects • Software libraries, frameworks
Application Class 3	Critical software, software products	• Mission-critical software • Software that is sold as a produt (with warranties) • Software that serves as research infrastructure

[Schlauch et al. 2018]
[Fritzsch 2023]

Potential risks, expected scope and lifetime determine the application class



[Schlauch et al. 2018]

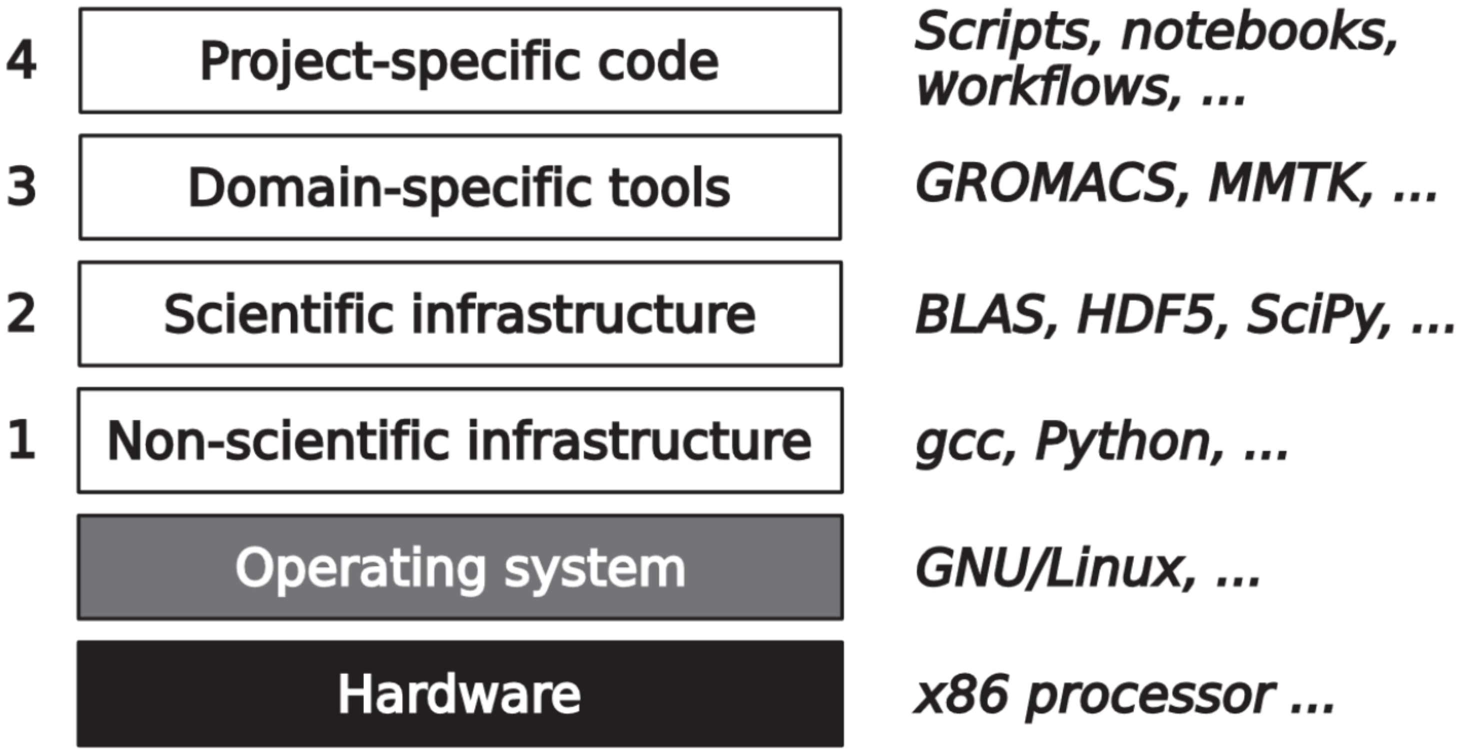
Additional Dimension: Categorization based on Criticality

- Safety-critical software
  - Failure results in loss of life, injury or damage to the environment;
  - Example: Railway interlocking system
- Mission-critical software
  - Failure results in failure of some goal-directed activity and/or loss of critical infrastructure;
  - Example: Spacecraft navigation system
- Business-critical software
  - Failure results in high economic losses or damage to reputation;
  - Example: Customer accounting system in a bank

⇒ Dependability

- Policy-critical software (?)

Additional Dimension (not yet included): Software Layers

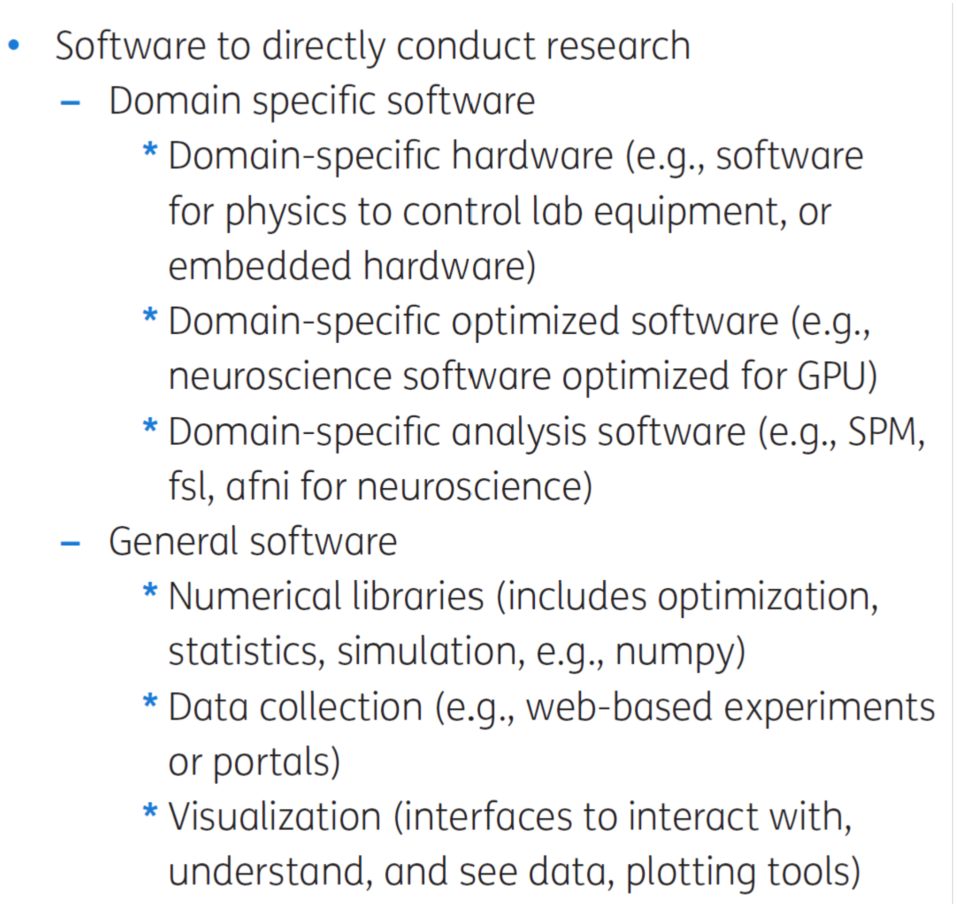


4	Project-specific code	*Scripts, notebooks, workflows, ...*
3	Domain-specific tools	*GROMACS, MMTK, ...*
2	Scientific infrastructure	*BLAS, HDF5, SciPy, ...*
1	Non-scientific infrastructure	*gcc, Python, ...*
	Operating system	*GNU/Linux, ...*
	Hardware	*x86 processor ...*

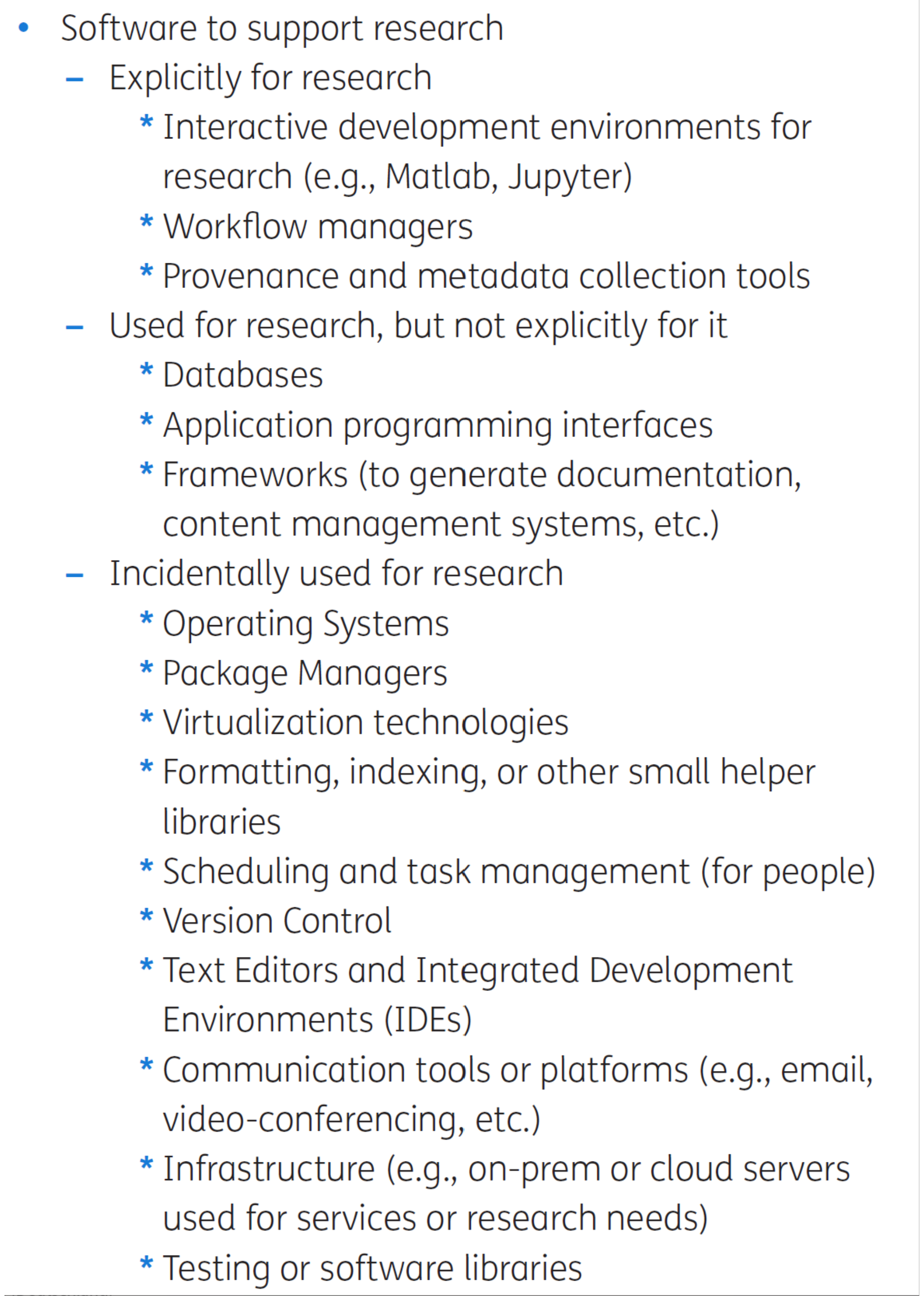
Figure 1. Typical scientific software stack.

[Hinsen 2019]

Related:The Research Software Encyclopedia´s Taxonomy



- Software to directly conduct research
  - Domain specific software
    - * Domain-specific hardware (e.g., software for physics to control lab equipment, or embedded hardware)
    - * Domain-specific optimized software (e.g., neuroscience software optimized for GPU)
    - * Domain-specific analysis software (e.g., SPM, fsl, afni for neuroscience)
  - General software
    - * Numerical libraries (includes optimization, statistics, simulation, e.g., numpy)
    - * Data collection (e.g., web-based experiments or portals)
    - * Visualization (interfaces to interact with, understand, and see data, plotting tools)



- Software to support research
  - Explicitly for research
    - * Interactive development environments for research (e.g., Matlab, Jupyter)
    - * Workflow managers
    - * Provenance and metadata collection tools
  - Used for research, but not explicitly for it
    - * Databases
    - * Application programming interfaces
    - * Frameworks (to generate documentation, content management systems, etc.)
  - Incidentally used for research
    - * Operating Systems
    - * Package Managers
    - * Virtualization technologies
    - * Formatting, indexing, or other small helper libraries
    - * Scheduling and task management (for people)
    - * Version Control
    - * Text Editors and Integrated Development Environments (IDEs)
    - * Communication tools or platforms (e.g., email, video-conferencing, etc.)
    - * Infrastructure (e.g., on-prem or cloud servers used for services or research needs)
    - * Testing or software libraries

[Sochat et al. 2022], https://rseng.github.io/rseng/

Research Software Examples

Example for Category 1.1 (Modeling and simulation): Modularization of Earth-system simulation software as basis for domain-specific languages

Software Modularization

How to

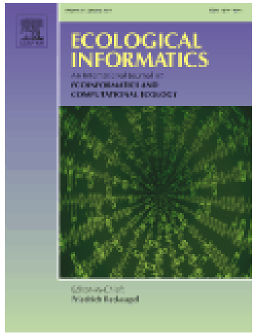
- improve maintainability, stability, reusability, reproducibility, … ?
- enable scalable execution in the Cloud?
- parallelize for high performance computing?
- test for higher quality?
- achieve higher flexibility?

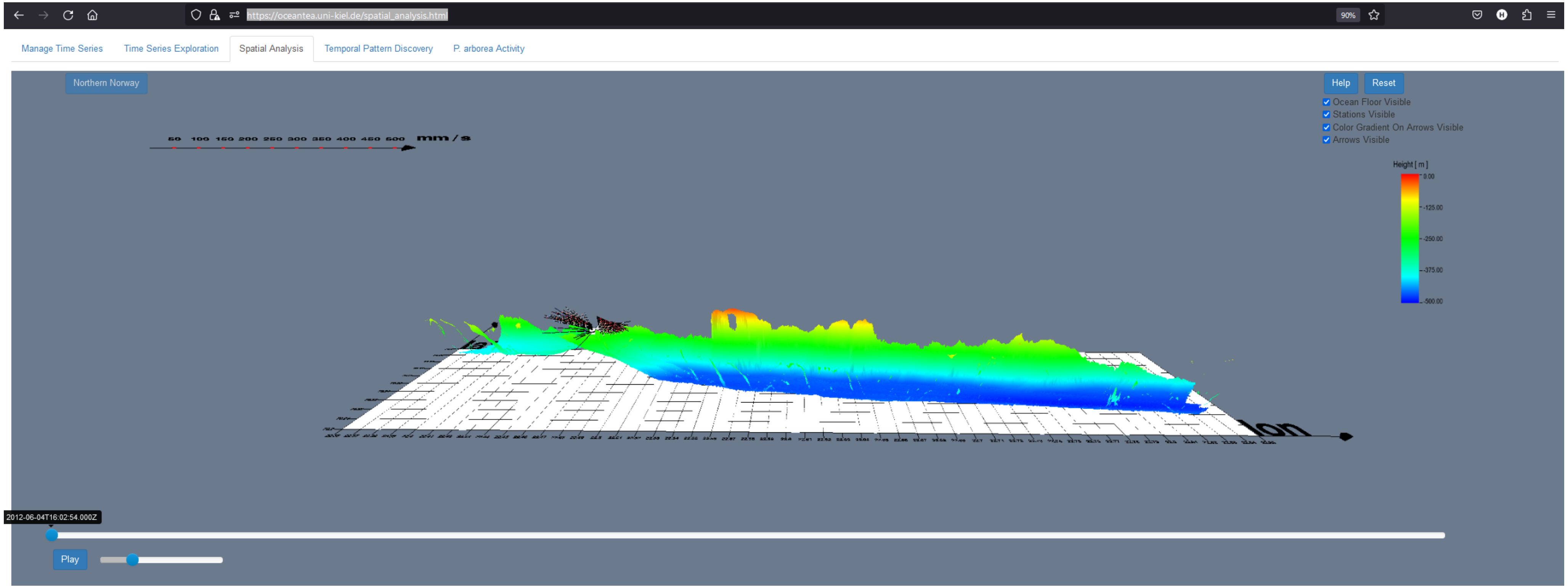
[Johanson & Hasselbring 2017, Jung et al. 2021, 2022a, 2022b]

OceanDSL

Funded by
DFG Deutsche Forschungsgemeinschaft
German Research Foundation

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





Paper on the analysis results: [Johanson et al. 2017]
Paper on the software architecture: [Johanson et al. 2016]
Code: https://github.com/cau-se/oceantea

future ocean
KIEL MARINE SCIENCES

Examples for Category 2 (Technology Research Software)

SustainKieker

Kieker

https://github.com/kieker-monitoring

Kieker: A monitoring framework for software engineering research
[Hasselbring and van Hoorn 2020]

ExplorViz

https://github.com/ExplorViz

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

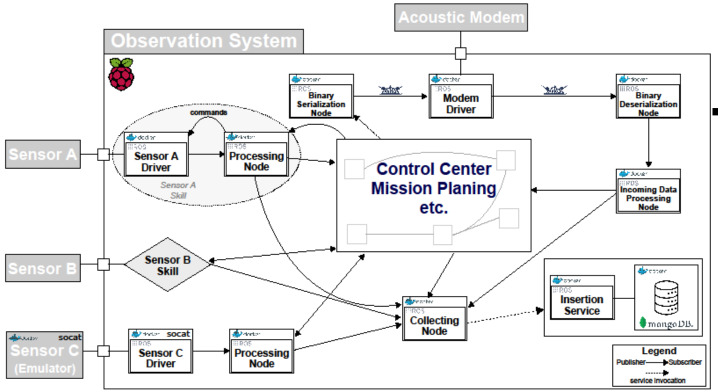
Theodolite

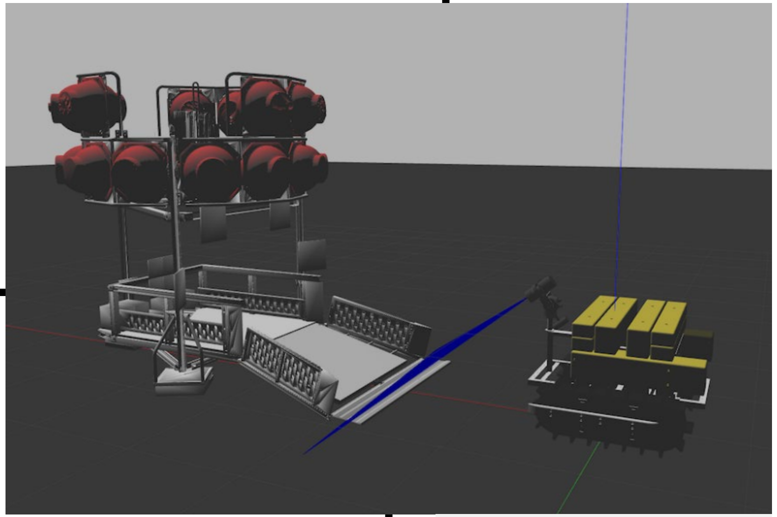
https://www.theodolite.rocks

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

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

Digital Twin Prototype





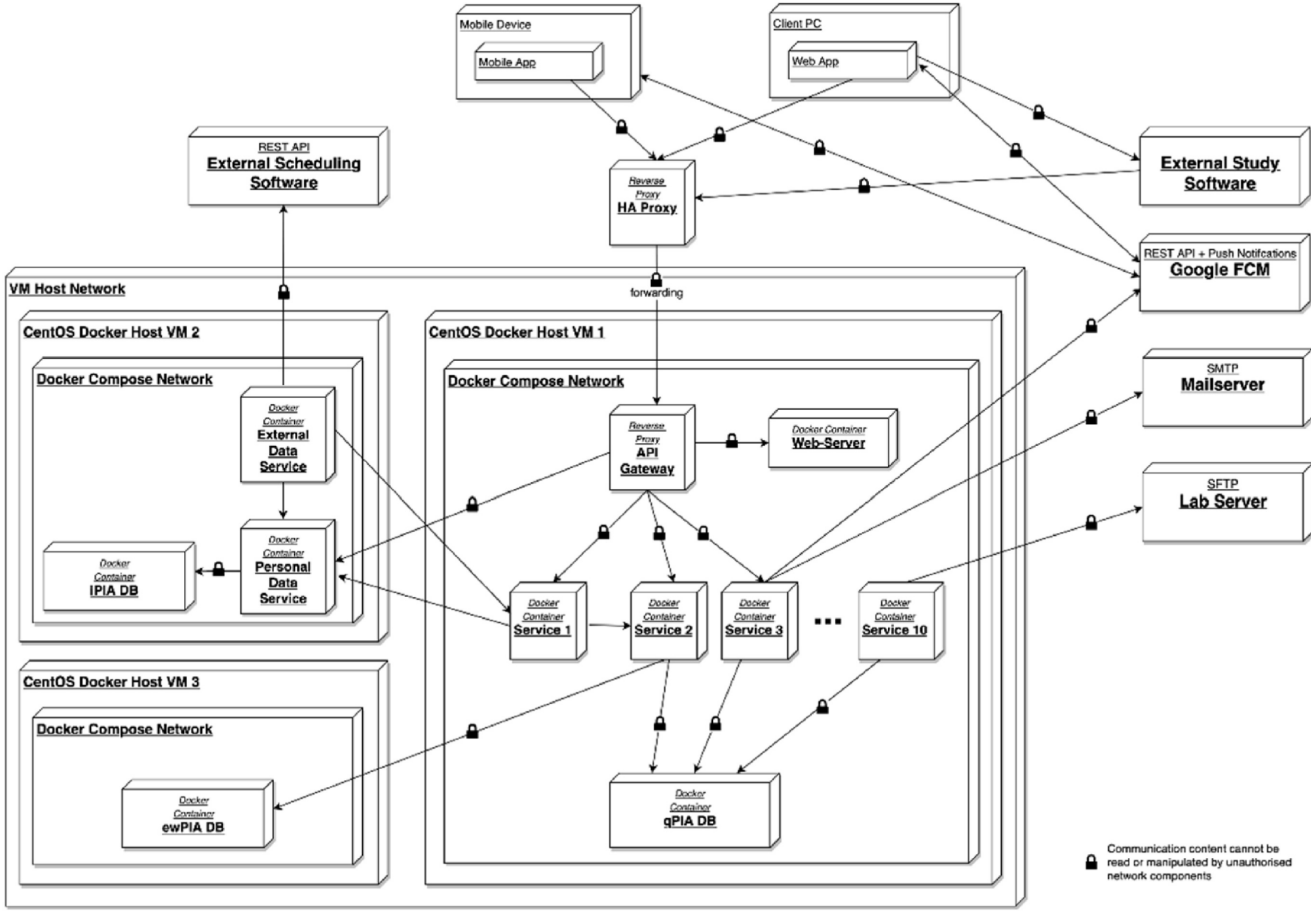
Digital Twin

Physical Twin

[Barbie et al. 2021]

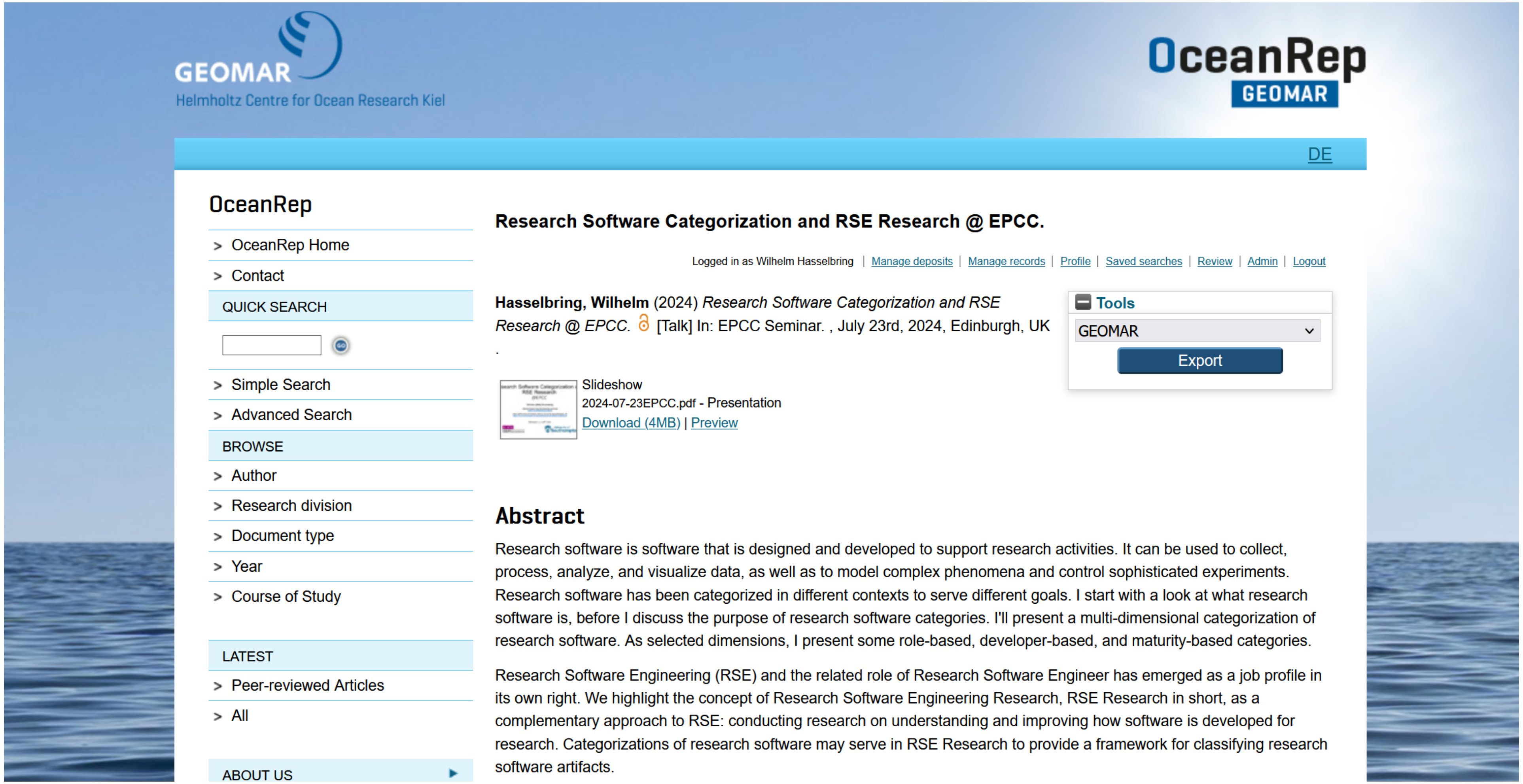
GEOMAR

Examples for Category 3.2 (Infrastructure): PIA: Data Collection for Medical Research



[Heise et al. 2022] https://github.com/hzi-braunschweig/pia-system

Examples for Category 3.6 and 3.8 (Infrastructure): EPrints Software for Open Access Repositories



Research Software Categorization and RSE Research @ EPCC.

Logged in as Wilhelm Hasselbring | Manage deposits | Manage records | Profile | Saved searches | Review | Admin | Logout

Hasselbring, Wilhelm (2024) *Research Software Categorization and RSE Research @ EPCC.* [Talk] In: EPCC Seminar. , July 23rd, 2024, Edinburgh, UK .

Slideshow
2024-07-23EPCC.pdf - Presentation
Download (4MB) | Preview

Abstract

Research software is software that is designed and developed to support research activities. It can be used to collect, process, analyze, and visualize data, as well as to model complex phenomena and control sophisticated experiments. Research software has been categorized in different contexts to serve different goals. I start with a look at what research software is, before I discuss the purpose of research software categories. I'll present a multi-dimensional categorization of research software. As selected dimensions, I present some role-based, developer-based, and maturity-based categories.

Research Software Engineering (RSE) and the related role of Research Software Engineer has emerged as a job profile in its own right. We highlight the concept of Research Software Engineering Research, RSE Research in short, as a complementary approach to RSE: conducting research on understanding and improving how software is developed for research. Categorizations of research software may serve in RSE Research to provide a framework for classifying research software artifacts.

https://oceanrep.geomar.de/id/eprint/60546/

Retrospect:
RSE (Meta) Research

RSE CON24

Newcastle, Sept 03-05, ’24

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]

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

[Claus et al. 2022] Claus, M., Gundlach, S., Hasselbring, W., Jung, R., Rath, W. und Schnoor, H.: “Modularizing Earth system models for interactive simulation.” Informatik Spektrum, 45. pp. 300-303. 2022, DOI https://doi.org/10.1007/s00287-022-01490-z

[Courbebaisse et al. 2023] G. Courbebaisse, B. Flemisch, K. Graf, U. Konrad, J. Maassen, R. Ritz: “Research Software Lifecycle”, Zenodo, 2023. DOI https://doi.org/10.5281/zenodo.8324828

[Deekshitha et al. 2024] Deekshitha, Bakhshi, R., Maassen, J., Martinez Ortiz, C., van Nieuwpoort, R., Jansen, S. : “RSMM: A Framework to Assess Maturity of Research Software Project”. arXiv e-prints. 2024, DOI https://doi.org/10.48550/arXiv.2406.01788

[Döll et al. 2023] Döll, P., Sester, M., Feuerhake, U., Frahm, H., Fritzsch, B., Hezel, D.C., Kaus, B., Kolditz, O., Linxweiler, J., Müller Schmied, H., Nyenah, E., Risse, B., Schielein, U., Schlauch, T., Streck, T., Van den Oord, G.: “Sustainable research software for high-quality computational research in the Earth System Sciences: Recommendations for universities, funders and the scientific community in Germany”. 2023, DOI https://doi.org/10.23689/fidgeo-5805

[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

[Gruenpeter et al. 2021] Gruenpeter, M., Katz, D. S., Lamprecht, A.-L., Honeyman, T., Garijo, D., Struck, A., Niehues, A., Martinez, P. A., Castro, L. J., Rabemanantsoa, T., Chue Hong, N. P., Martinez-Ortiz, C., Sesink, L., Liffers, M., Fouilloux, A. C., Erdmann, C., Peroni, S., Martinez Lavanchy, P., Todorov, I., Sinha, M.: “Defining Research Software: a controversial discussion”. 2021. DOI https://doi.org/10.5281/zenodo.5504016

[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

References

[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., Lamprecht, A. L., Linxweiler, J., Rumpe, B.: “Toward Research Software Categories”. arXiv e-prints . DOI https://doi.org/10.48550/arXiv.2404.14364

[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

[Heise et al. 2022] J.-K. Heise, R. Dey, M. Emmerich, Y. Kemmling, S. Sistig, G. Krause, S. Castell: “Putting digital epidemiology into practice: PIA – prospective monitoring and management application,” Informatics in Medicine Unlocked, vol. 30, p. 100931, 2022. DOI https://doi.org/10.1016/j.imu.2022.100931

[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

[Hinsen 2019] K. Hinsen, “Dealing With Software Collapse,” Computing in Science Engineering, vol. 21, no. 3, pp. 104 108, May 2019, DOI 10.1109/MCSE.2019.2900945

[Johanson et al. 2016] 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. 2017] 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

References

[Jung et al. 2021] R. Jung, S. Gundlach, S. Simonov, W. Hasselbring: “Developing Domain-Specific Languages for Ocean Modeling”. In: Software Engineering 2021 Satellite Events, http://ceur-ws.org/Vol-2814/

[Jung et al. 2022a] R. Jung, S. Gundlach, W. Hasselbring: “Software development processes in ocean system modeling.” In: International Journal of Modeling, Simulation, and Scientific Computing, 13 (02). 2022, DOI https://doi.org/10.1142/S1793962322300023.

[Jung et al. 2022b] R. Jung, S. Gundlach, W. Hasselbring: “Thematic domain analysis for ocean modeling.” In: Environmental Modelling & Software, 150. p. 105323. 2022, DOI https://doi.org/10.1016/j.envsoft.2022.105323.

[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

[Lamprecht et al. 2022] A.-L. Lamprecht et al.: “What Do We (Not) Know About Research Software Engineering?.” In: Journal of Open Research Software, 10(1). 2022. DOI https://doi.org/10.5334/jors.384

[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

[Sochat et al. 2022] Sochat, V., May, N., Cosden, I., Martinez-Ortiz, C. and Bartholomew, S., 2022. The Research Software Encyclopedia: A Community Framework to Define Research Software. Journal of Open Research Software, 10(1). DOI https://doi.org/10.5334/jors.359

[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, 2023. DOI https://doi.org/10.54900/xdh2x-kj281

[zu Castell et al. 2024] W. zu Castell, D. Dransch, G. Juckeland, M. Meistring, B. Fritzsch, R. Gey, B. Höpfner, M. Köhler, C. Meeßen, H. Mehrtens, F. Mühlbauer, S. Schindler, T. Schnicke, R. Bertelmann: “Towards a Quality Indicator for Research Data publications and Research Software publications -- A vision from the Helmholtz Association”. 2024, DOI https://doi.org/10.48550/arXiv.2401.08804