CATEGORIZATIONS OF RESEARCH SOFTWARE

Stephan Druskat[1], Jan Linxweiler[2], Sören Peters[2]
[1]German Aerospace Center (DLR), Institute for Software Technology
[2]Technische Universität Braunschweig

DOI 10.5281/zenodo.10159169

Overview

- *Recap:* Why classify research software?
- Research software definition & categorization
- *Preview:* Dimensions of software categorization
- Examples of categorizations
- Conclusion

Research software definition & categorization

- Definition of research software impacts categorization efforts

- Prescriptive definition vs. descriptive definition
  - **Prescriptive** [1]: Created during research process or for research purpose
  - **Descriptive**: Used during research process
  - **Mixed** [2]: Used during research process or research object itself

- Basic question (one removed): What is the purpose of the definition?

[1] M. Gruenpeter et al., "Defining Research Software: a controversial discussion," Zenodo, Sep. 2021. doi: 10.5281/zenodo.5504016.
[2] W. Hasselbring, L. Carr, S. Hettrick, H. Packer, and T. Tiropanis, "From FAIR research data toward FAIR and open research software," it - Information Technology, vol. 62, no. 1, pp. 39–47, Feb. 2020, doi: 10.1515/itit-2019-0040.

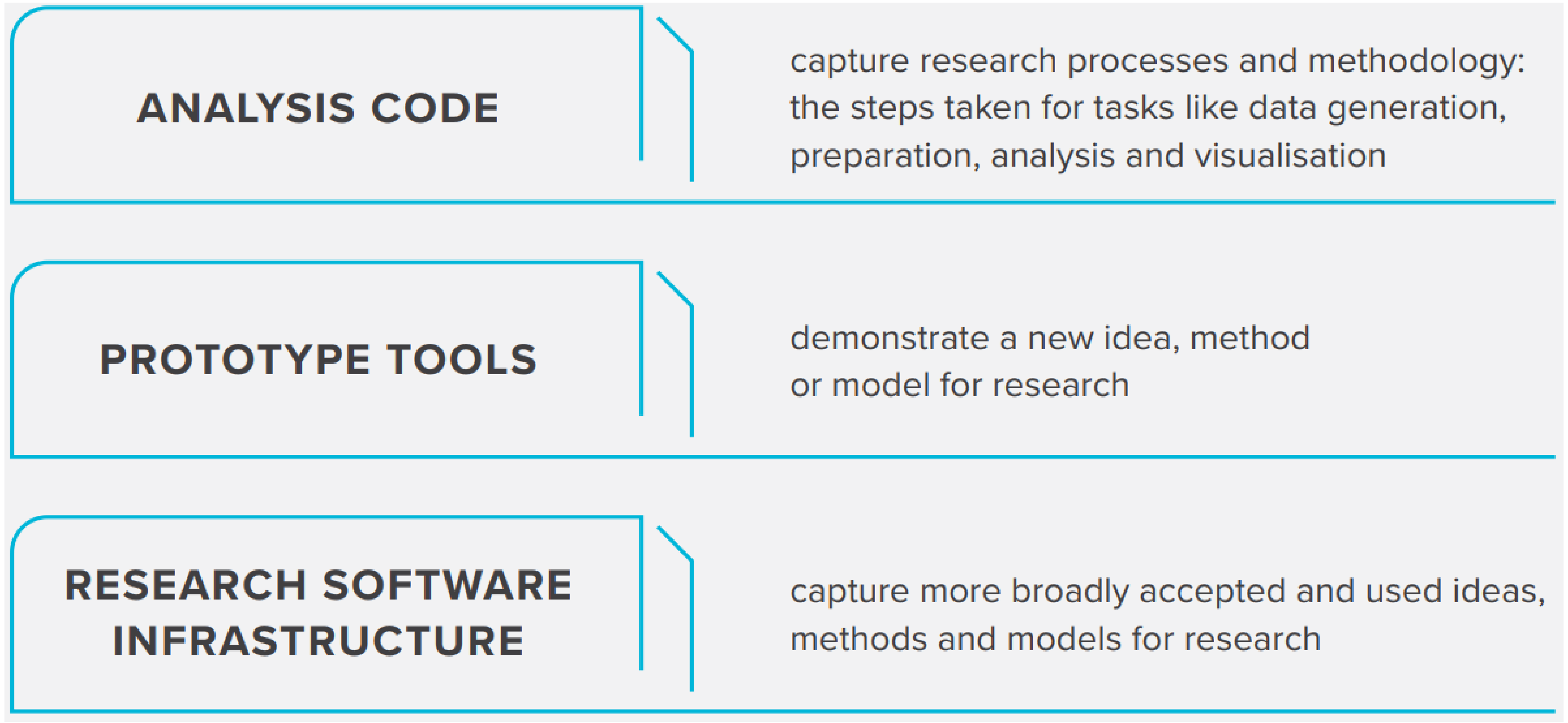
Categorization dimensions

1. Scope
2. Purpose
3. Categorization context
4. Category properties
5. Consequences for software creation
6. Consequences for software use
7. Inter-categorial relations

→ See also B. Rumpe’s presentation after the coffee break

CATEGORIZATION EXAMPLES

ARDC National Agenda for Research Software



ANALYSIS CODE

capture research processes and methodology: the steps taken for tasks like data generation, preparation, analysis and visualisation

PROTOTYPE TOOLS

demonstrate a new idea, method or model for research

RESEARCH SOFTWARE INFRASTRUCTURE

capture more broadly accepted and used ideas, methods and models for research

[3] Australian Research Data Commons, "A National Agenda for Research Software," Mar. 2022, doi: 10.5281/zenodo.6378082.

ARDC National Agenda for Research Software

Dimension	Value
Scope	Increasing software recognition
Purpose	Describing *software purpose*
Categorization context	Research software policy
Category properties	Different challenges w.r.t. recognition
Software creation	FAIR software > high quality software > sustainable (maintained) software
Software use	(Implicit)
Inter-categorial relations	Dependency, transitive value

Hinsen (2019)

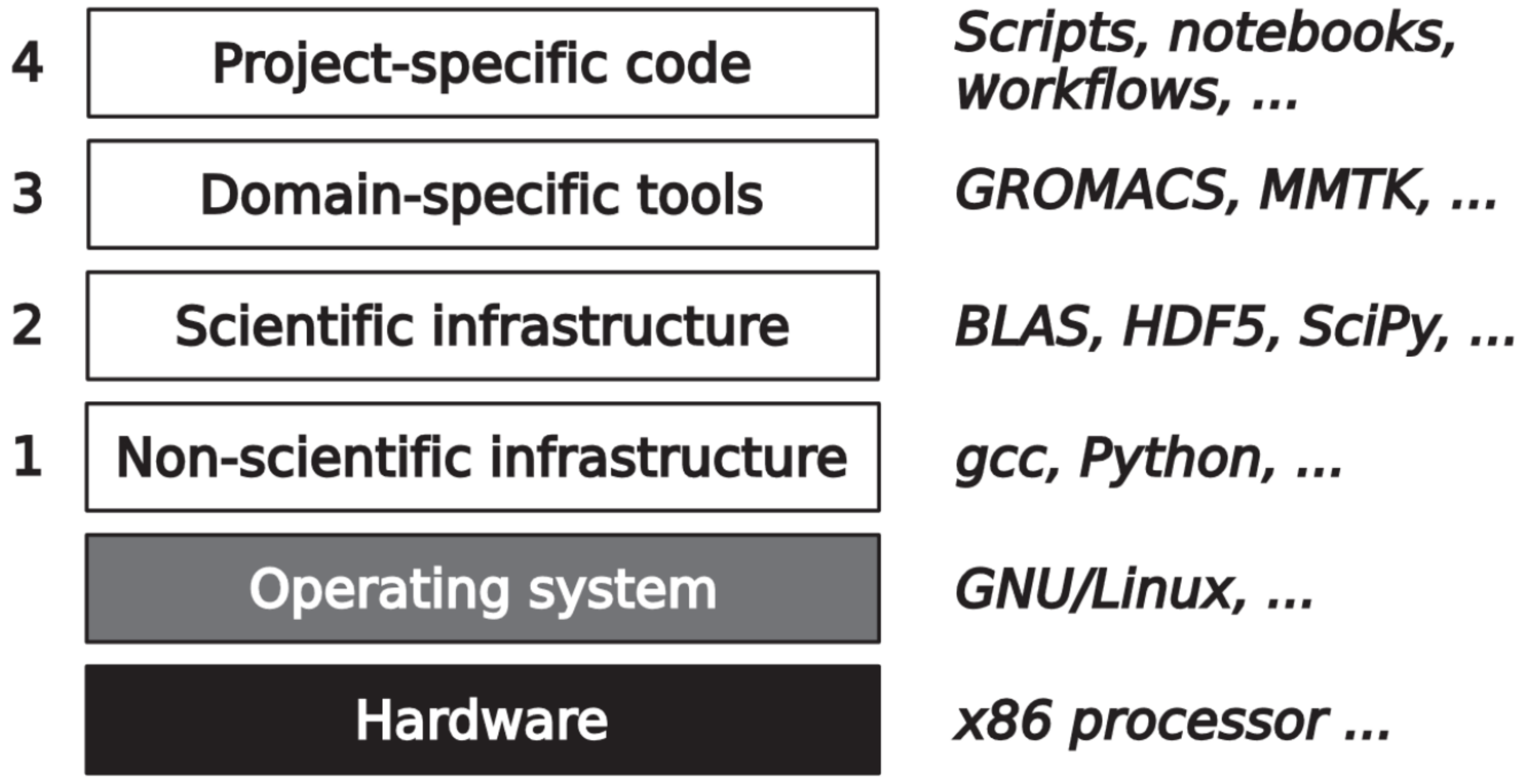


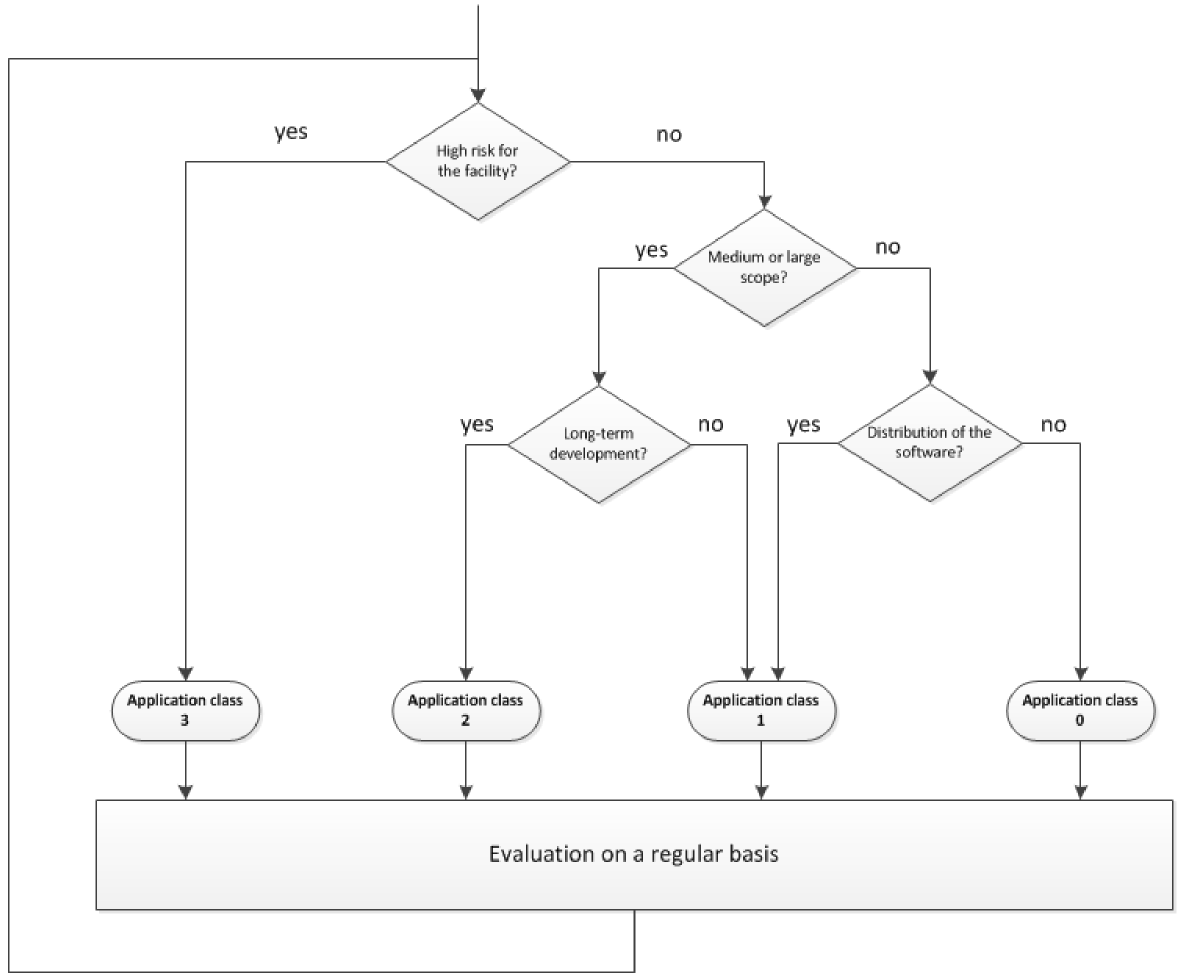
Figure 1. Typical scientific software stack.

[4] 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.

Hinsen (2019)

Dimension	Value
Scope	Describing principles of software collapse
Purpose	Identify dependent layers of different (academic) specificity to model threat
Categorization context	Research software sustainability
Category properties	Domain specificity
Software creation	Build on stable lower layers, quickly react to threats, accept agility
Software use	Decreasing specificity of application domain from top to bottom
Inter-categorial relations	Dependency, transitive threats

DLR Application Classes

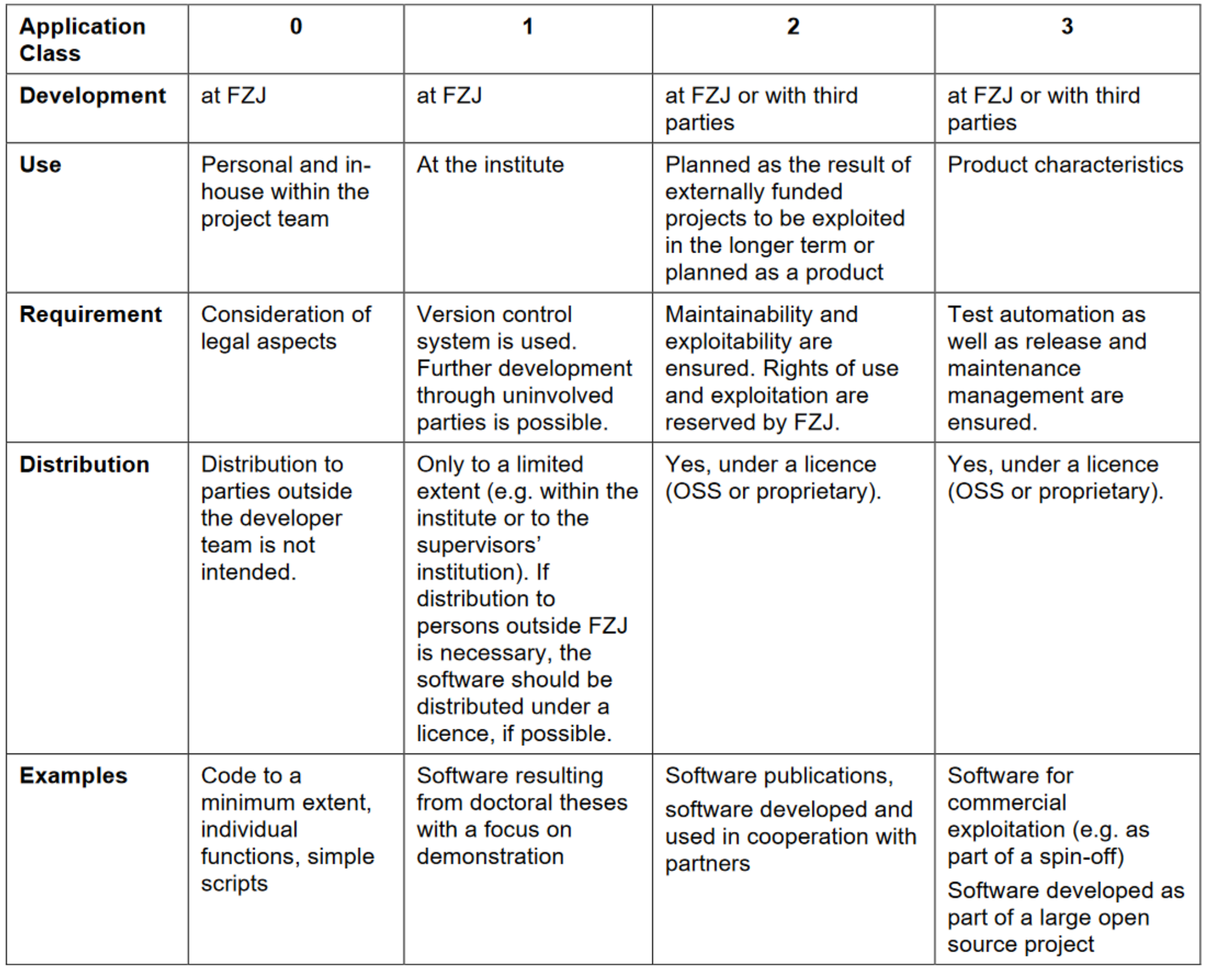


[5] Schlauch, Tobias, Meinel, Michael, and Haupt, Carina, “DLR Software Engineering Guidelines,” Zenodo, Aug. 2018. doi: 10.5281/zenodo.1344612.

DLR Application Classes

Dimension	Value
Scope	Guidelines for software engineering at an academic institution
Purpose	Identify suitable quality requirements
Categorization context	Institutional policy and practice
Category properties	Criticality, institutional risk, projected use, development timeline, distribution
Software creation	Increasingly employ (formalized) software engineering methods
Software use	Increased (critical) use by increasingly large community
Inter-categorial relations	Transitive requirements

FZJ Application Classes



Application Class	0	1	2	3
Development	at FZJ	at FZJ	at FZJ or with third parties	at FZJ or with third parties
Use	Personal and in-house within the project team	At the institute	Planned as the result of externally funded projects to be exploited in the longer term or planned as a product	Product characteristics
Requirement	Consideration of legal aspects	Version control system is used. Further development through uninvolved parties is possible.	Maintainability and exploitability are ensured. Rights of use and exploitation are reserved by FZJ.	Test automation as well as release and maintenance management are ensured.
Distribution	Distribution to parties outside the developer team is not intended.	Only to a limited extent (e.g. within the institute or to the supervisors' institution). If distribution to persons outside FZJ is necessary, the software should be distributed under a licence, if possible.	Yes, under a licence (OSS or proprietary).	Yes, under a licence (OSS or proprietary).
Examples	Code to a minimum extent, individual functions, simple scripts	Software resulting from doctoral theses with a focus on demonstration	Software publications, software developed and used in cooperation with partners	Software for commercial exploitation (e.g. as part of a spin-off) Software developed as part of a large open source project

[6] O. Bertuch, D. Oliveira, U. Schelhaas, and A. Storm, "Guidelines for the development and distribution of software at Forschungszentrum Jülich," 2022. [Online]. Available: http://hdl.handle.net/2128/33259

FZJ Application Classes

Dimension	Value
Scope	Guidelines for software engineering at an academic institution
Purpose	Identify suitable quality requirements
Categorization context	Institutional policy and practice
Category properties	Commercial exploitation, projected usage, development timeline, distribution
Software creation	Increasingly employ good software engineering practice
Software use	Increased use by increasingly large community, towards product status
Inter-categorial relations	Transitive requirements, legal requirements

Relation to generic categorizations?

- W3ID software type classes:
  - `CommandLineApplication`
  - `DesktopApplication`
  - `NotebookApplication`
  - `ServerApplication`
  - `SoftwareImage`
  - `SoftwareLibrary`
  - `SoftwarePackage`
  - `TerminalApplication`

[7] M. van Gompel and D. Garijo, “Software types”, [Online]. Available: https://w3id.org/software-types

Conclusion

- Categorization relies on a stable definition of research software
- Existing categorizations (may) differ in relevant dimensions
- Should the relation to generic categorizations be considered? (Probably not.)

- **Proposal: Our work should be able to represent the relevant dimensions of existing categorizations, i.e., be a superset**

Draft mapping proposal

Resarch Software	Hinsen Stack	DLR Class	Risk	Scope	Distribution	Longevity	ARDC Categories	RSER Draft Categories
True	Project-specific code	3-0	high-none	large-singular	wide-none	long-one-time	Analysis, Prototype	MSDA, PCOP
True	Domain-specific tools	3-2	high-medium	large-medium	wide-considerable	long-medium	Prototype, Infrastructure	MSDA, POCP, IPS?
True / False	Scientific infrastructure	3-2	high-medium	large-medium	wide-considerable	long-medium	Infrastructure	ECS, IPS
False	Non-scientific infrastructure							ECS?
False	OS							
False	Hardware							

Questions?

- stephan.druskat@dlr.de
- @sdruskat@fosstodon.org