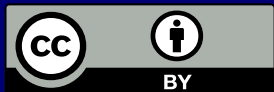


The State of Research Software Development at Paul Scherrer Institute

Results and Discussion
from a 2025 Staff Survey



Michael P. Weinold, Achim Gsell and Elsa Germann
Villigen, November 2025



doi:XXXXXXXXXX

Executive Summary

This is the first publicly available staff survey focusing on the state of research software engineering (RSE) at Paul Scherrer Institute (PSI). 56 participants responded, each having spent more than half an hour on the survey on average.

Key findings:

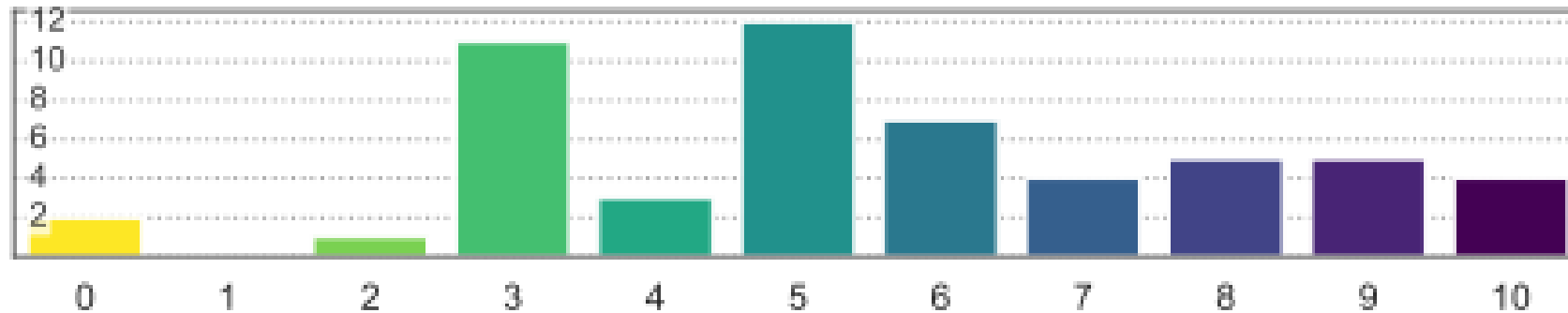
- Staff on permanent contracts consistently have more experience in software development, but in general spend less time on it.
- A large share of overall software development efforts are undertaken by staff on temporary contracts (PhD students, etc.).
- Python is the most widely used programming language, followed by C/C++ and MATLAB.
- Only 5% of respondents indicate that on-the-job training has helped them improve their software development skills. 80% of respondents therefore ask for workshops/trainings to be offered.
- Familiarity with widely adopted best-practices for research software development (“FAIR4RS”) is low, with only 16% of respondents implementing them.

Survey Participants

Basic Statistics



Researcher
without
Coding



Coder
without
Research

Participant responses on placing themselves on the “RSE Spectrum”

Responses were at an average of ~5 with a standard deviation of ~2.5

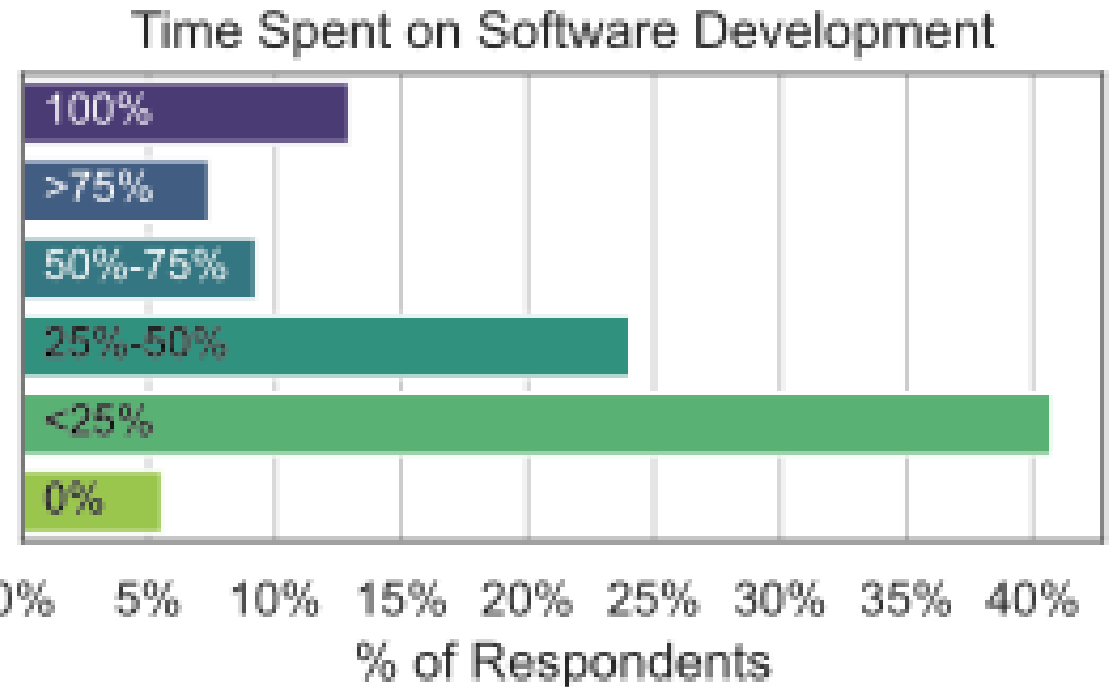
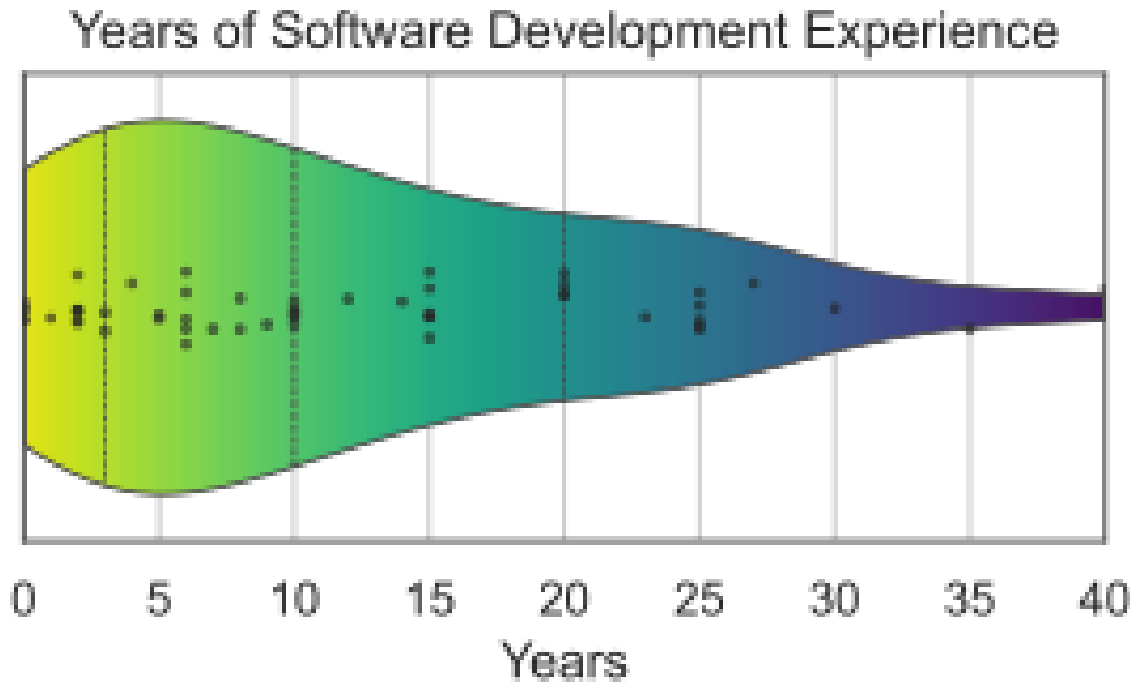
54 participants filled out our survey on research software development practices sent out in September 2025. The survey remained open for 2 months. The **average time spent on the survey was 36 minutes**. 30% definitely identify themselves as “research software engineers”, 30% do not and 30% are unsure.

See Also:

<https://rse.swiss/about/>
Cohen et al. (2020), [doi:gntv96](https://doi.org/10.1016/j.gntv.2020.100096)

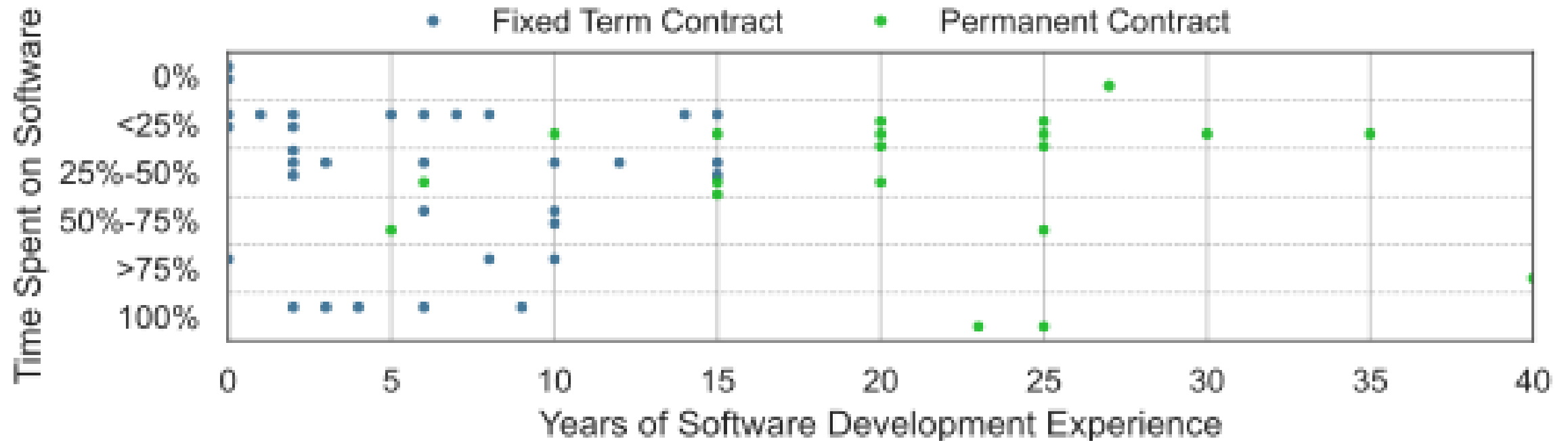
Survey Participants

Software Development Background



Respondents on average have ~11.7 years of software development experience. Only ~13% of respondents spend all their work time on software development, while **the majority is spending a quarter or less of their work time on software development.** Note that many IT-related duties, such as “system administration” are not explicitly represented in these responses.

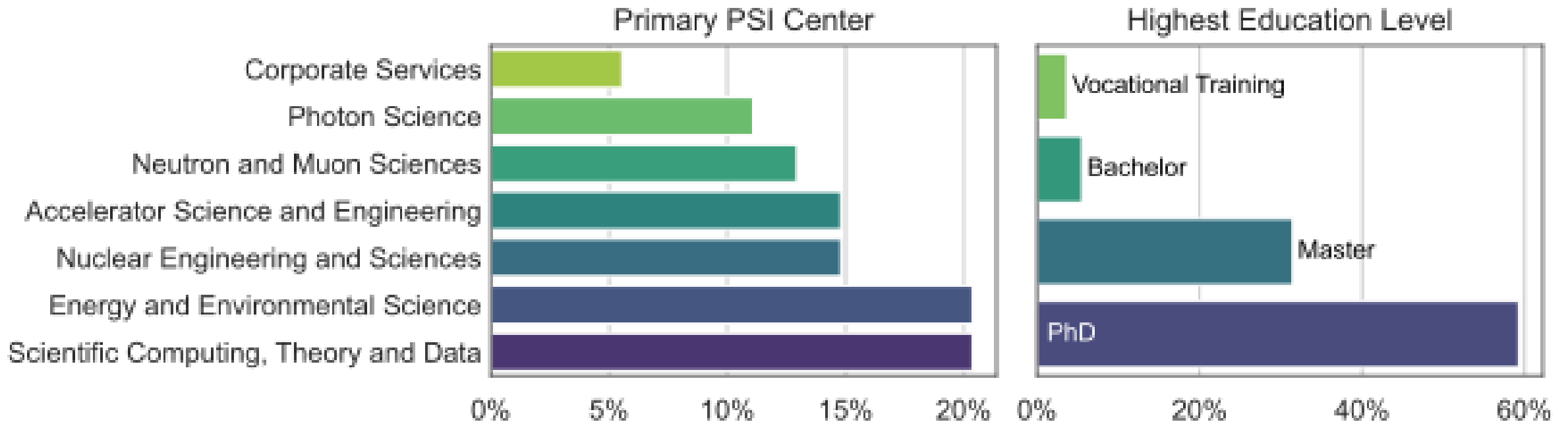
Software Development Background by Contract Type



Of the respondents, **those on permanent contracts consistently had more experience in software development**, although no clear trend can be established of those respondents also spending more writing software as part of their work. This is likely due to the **additional managerial tasks associated with higher levels of seniority and permanent positions**.

Survey Participants

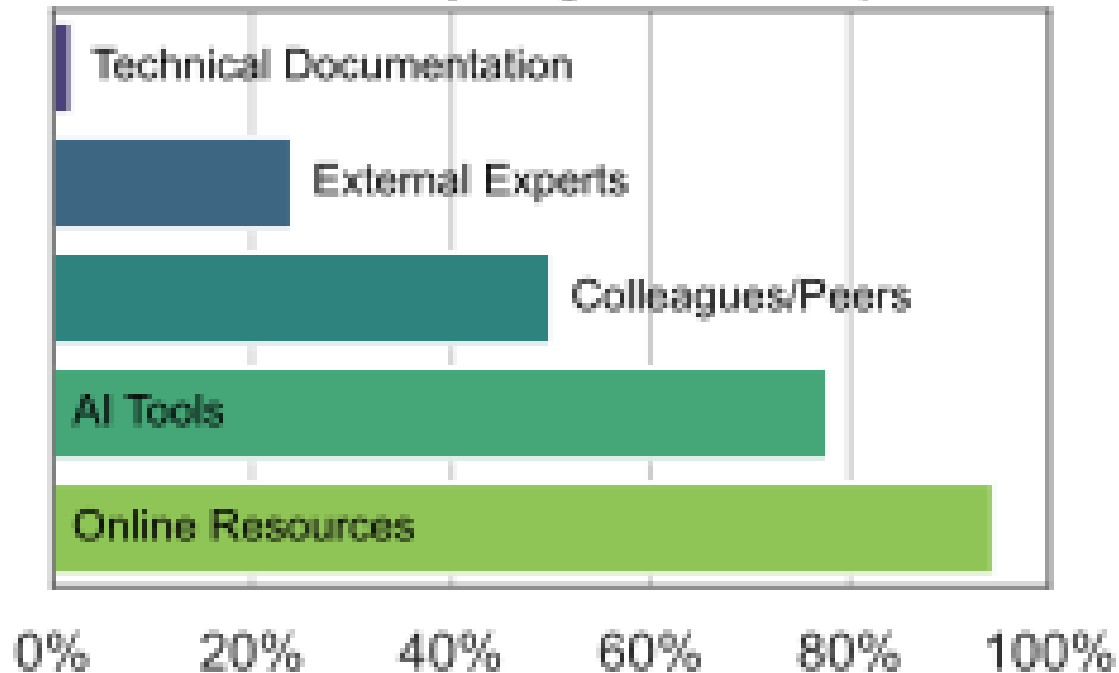
Corporate and Educational Background



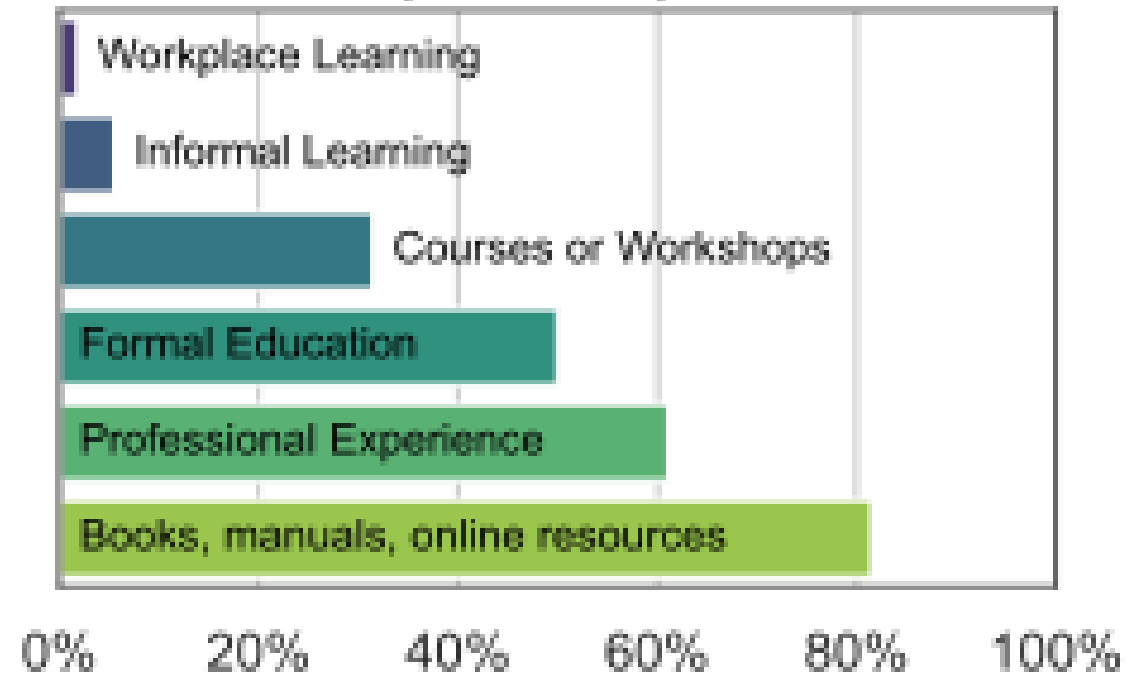
All centers of Paul Scherrer Institute (PSI) are represented in the survey, with the highest number of responders being affiliated EES and CSD at ~20% each. Nearly 60% of responders have obtained a doctorate or equivalent as their highest level of education.

Software Development Skills Training and Learning

Where do you go to for help?



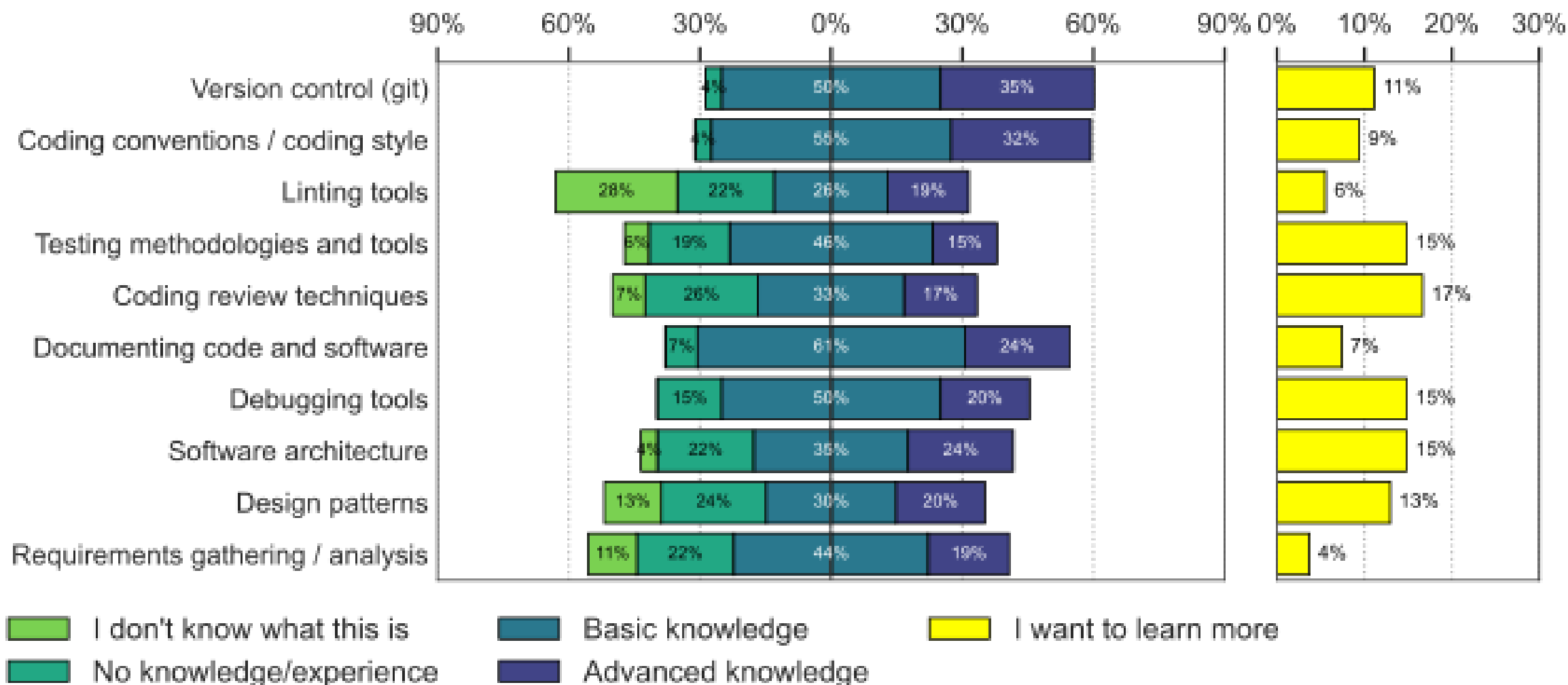
How did you learn your skills?



AI tools already come in second place in the list of resources which respondents go to for help. **Notably, only a <5% of respondents mention "workplace learning" as the primary source of their software development skills.** Results on other slides clearly show a desire for this to improve.

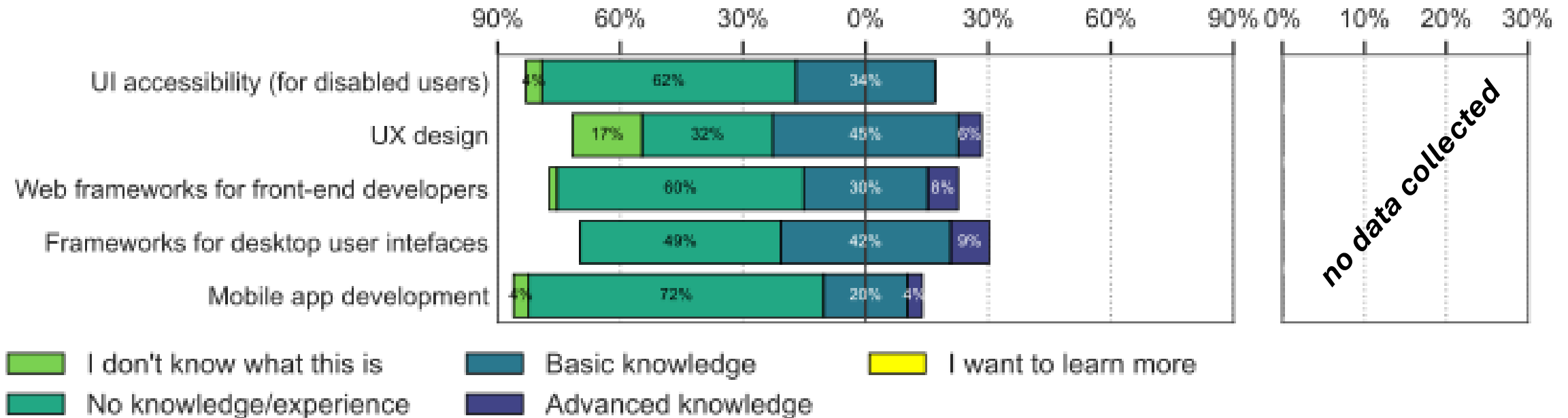
Software Development Skills

Technical Aspects of Development



Software Development Skills

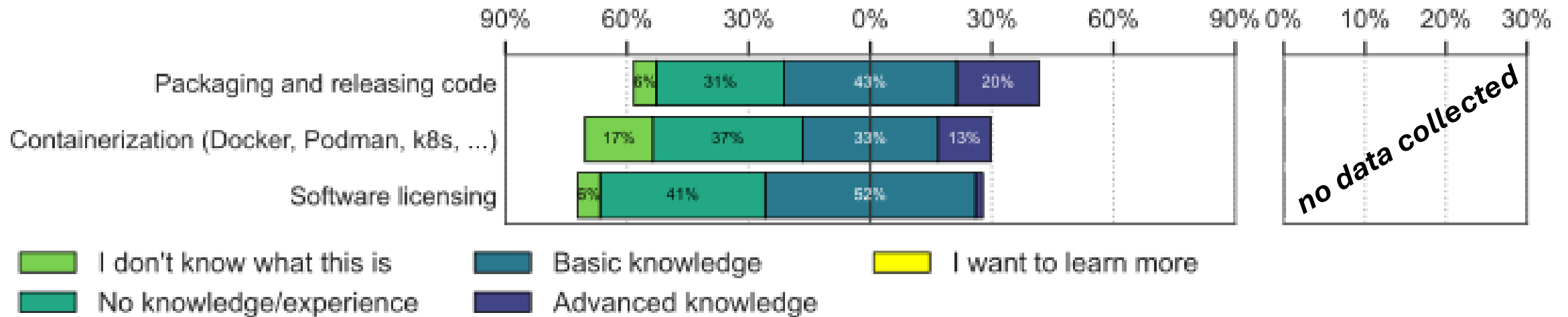
Software User Interfaces



The comparison of participant skill self-assessment between technical development aspects (previous slide) and software user interfaces confirms that **participants have little experience in user-interface/user-experience development**, reflecting the highly specific nature of the tools developed, which are often accessed only via Python programming interfaces.

Software Development Skills

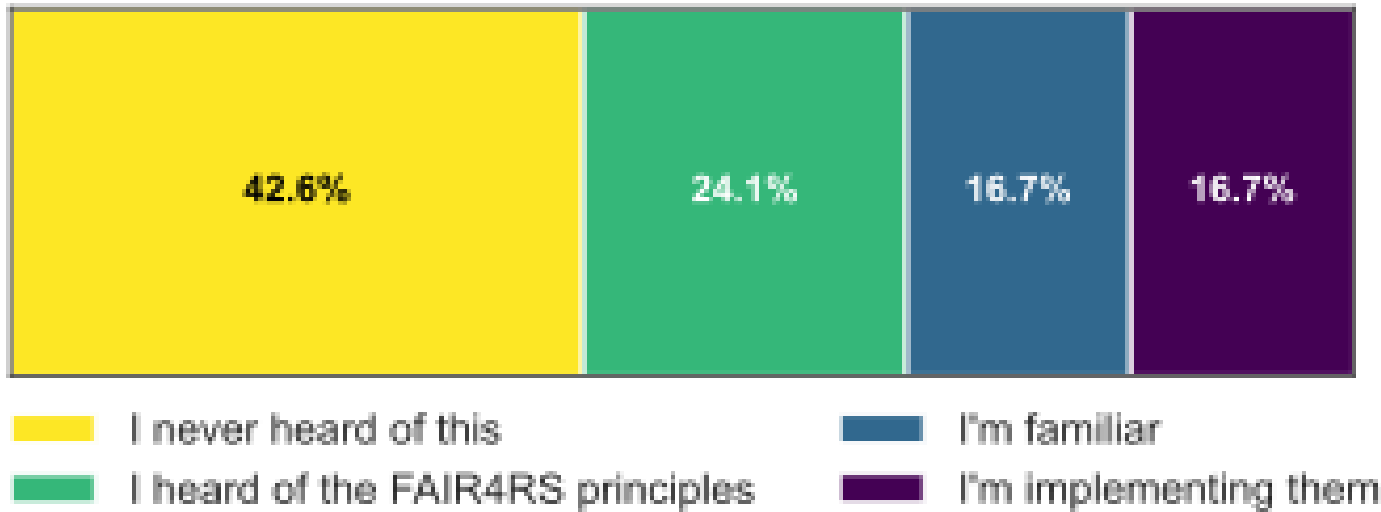
Software Deployment and Distribution



Similarly, **few participants have experience packaging their software, which is a prerequisite for wider release and platform-independent distribution.** With PSI IT now offering on-premises hosting for containerized applications, staff training should on this method could improve the uptake of this offering and reduce the system administration workload for current software owners.

Software Development at PSI

Familiarity with “FAIR4RS” Best Practices



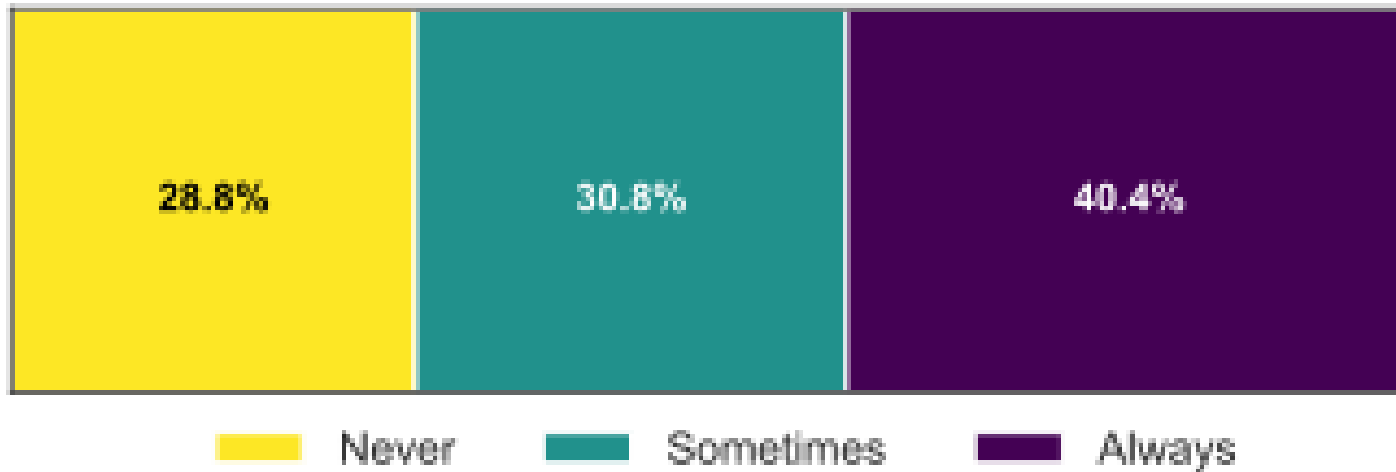
The “FAIR4RS Principles” are a set of guidelines that can help ensure the development of **F**indable, **A**ccessible, **I**nteroperable and **R**eusable research software. They are an *“provide an umbrella framework that integrates aspects of (...) existing efforts [in software and data discoverability]”* – [Barker et al. \(2022\)](#)

See Also:

Jense & Katz (2025), [doi:qckz](#)
Baker et al. (2022), [doi:gsjk4h](#)
Hong et al. (2022), [doi:gpt8ts](#)

Software Development at PSI

Familiarity with Open-Source Software Licenses



The Open-Source Initiative (OSI) is a global non-profit organization involved in advocacy for non-proprietary software. It maintains a list open-source software licenses. This approval provides a legal guarantee that the software source code is accessible and that users have the freedom to use, modify, and share the software and its derivatives.

See Also:

<https://opensource.org/licenses>
Meloca et al. (2018), [doi:g8hmb9](https://doi.org/10.1016/j.jss.2018.08.009)

Software Development at PSI

Used Programming Languages



Language	% of selections	% of respondents	# of respondents
Python	29.6%	92.6%	50
Bash / Shell	12.4%	38.9%	21
C++	11.2%	35.2%	19
C	8.9%	27.8%	15
JS/TypeScript	7.1%	22.2%	12
Matlab	4.7%	14.8%	8
Fortran	4.1%	13.0%	7
R	4.1%	13.0%	7
Rust	3.0%	9.3%	5
Julia	1.8%	5.6%	3
Node.js	1.8%	5.6%	3
C#	1.8%	5.6%	3
GAMS	1.2%	3.7%	2
Java	1.2%	3.7%	2
Go	1.2%	3.7%	2
HTML	0.6%	1.9%	1
WebAssembly	0.6%	1.9%	1
vimscript	0.6%	1.9%	1
EPICS Framework	0.6%	1.9%	1
SCLand LabView	0.6%	1.9%	1
Tcl/Tk	0.6%	1.9%	1
LabVIEW	0.6%	1.9%	1
IDL	0.6%	1.9%	1
Igor Pro	0.6%	1.9%	1
Mathematica	0.6%	1.9%	1

Correlation of Programming Languages and Centers

Accelerator Science and Engineering

Corporate Services

Energy and Environmental Science

Neutron and Muon Sciences

Nuclear Engineering and Sciences

Photon Science

Scientific Computing, Theory and Data

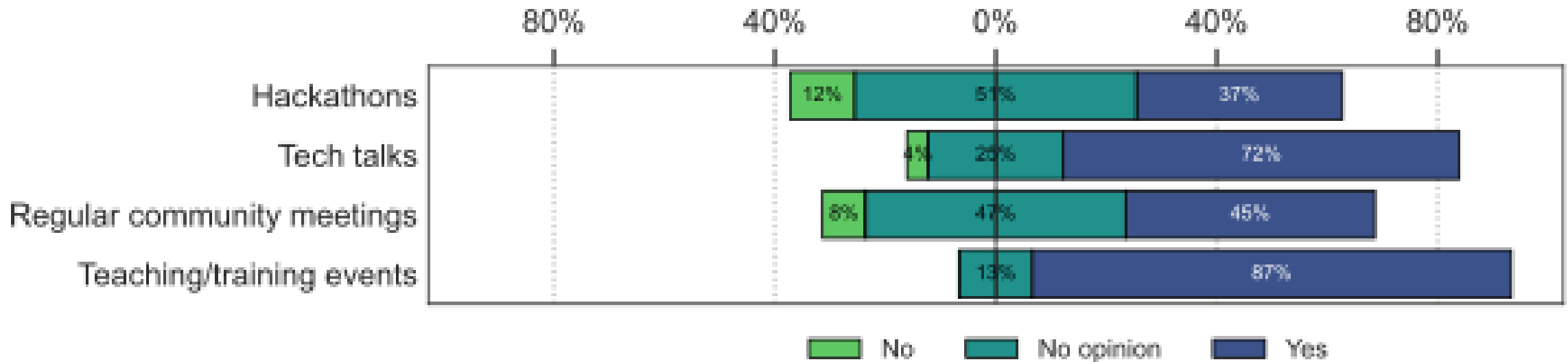
Python is the widely used language across all centers, with high-performance compiled languages such as C/C++ used predominantly in accelerator science.

Bash / Shell	C	C#	C++	EPICS Framework	Fortran	GAMS	Go	IDL (not very often)	Igor Pro	Java	Javascript / Typescript	Julia	LabVIEW	Mathematica	Matlab	Node.js	Python	R	Rust	SCL (for PLC) and Labview	Tcl/Tk	html	vimscript	webassembly
4	4	0	5	1	0	0	0	0	0	0	0	0	0	0	2	0	6	0	1	0	1	0	0	0
1	1	0	2	0	1	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0
1	0	2	1	0	1	1	0	0	0	0	1	0	1	1	3	0	10	4	0	1	0	0	0	0
1	4	1	4	0	2	0	0	1	0	0	3	1	0	0	2	1	7	0	0	0	0	1	0	0
2	1	0	2	0	1	1	0	0	0	0	1	0	0	0	1	0	8	2	0	0	0	0	0	0
4	1	0	3	0	1	0	0	0	1	0	1	0	0	0	0	0	6	0	1	0	0	0	0	0
8	4	0	2	0	1	0	2	0	0	2	6	2	0	0	0	2	10	1	3	0	0	0	1	1

Python is the widely used language across all centers, with high-performance compiled languages such as C/C++ used predominantly in accelerator science.

The Future of the Community at PSI

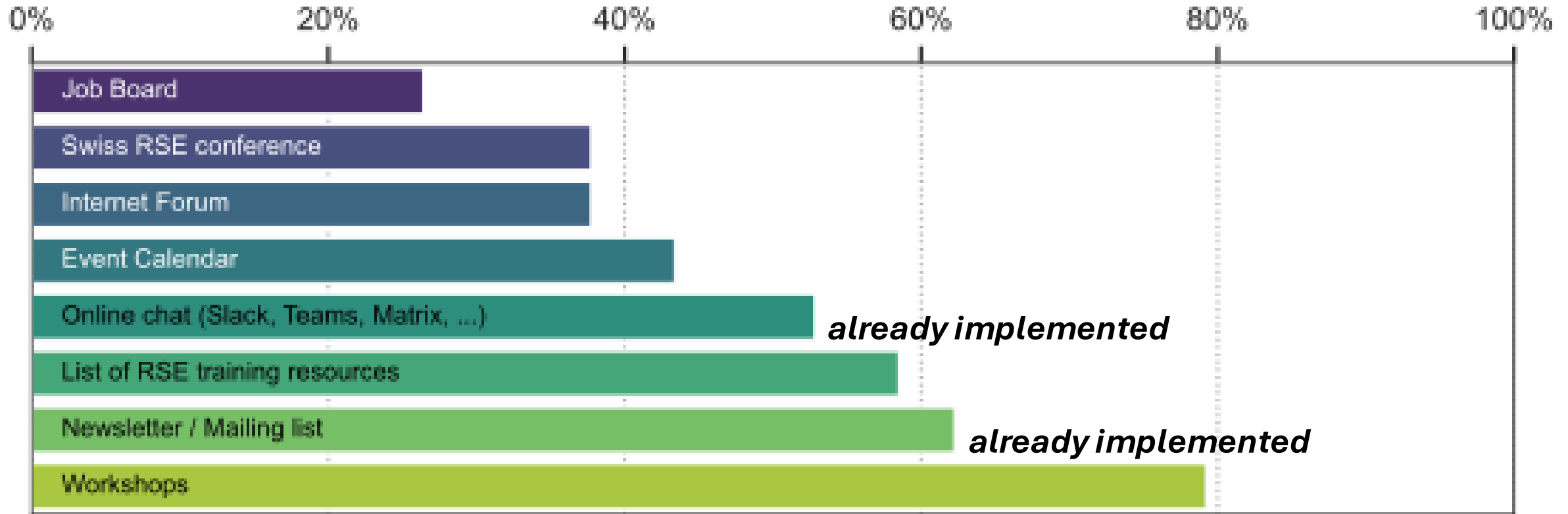
Interest in Different Events



Community feedback on possible future events has been positive, with **the most highly demanded event category being teaching/training events**. This is in line with responses indicating that currently <5% of software development skills have been acquired through workspace training. **A recurring RSE meeting series** including informal discussion and short “tech talk” presentations **was kicked off in November 2025**.

The Future of the Community at PSI

Interest in Community Services



Of all community services suggested in the survey, **additional workplace training is again the most highly requested**. This confirms responses on Slide 8 showing that **currently <5% of respondents learn software development skills in formal training at work**.

Voices from the Community

“I am mostly a manager and neither a hard-core code developer nor a scientist. (...) Getting the line management on board in as many PSI Centers as possible as sponsors or "supporters" might be important. Maybe even get into the official IT Strategy?”

“PSI should allow for more money for software engineers. Personally, I learned how to tackle code, but I'll never become a great coder, and the things I want to do are always more complex than what my coding skills can handle. Still, a mentoring system or teaching/training events could help.”

“Some dedicated time for training resources would be nice - so that this doesn't have to be apart from work. This would need buy in from management that RSE related learning / training activities are essential and count as work for those attending.”

Conclusion

Next Steps



The Research Software Engineering Community, based on this survey, will take these next steps:

- Hosting monthly (hybrid) community meetings, including “tech talks”, “show your tools”, etc.
- Raising awareness with management about:
 - ...the large share of often mission-critical software which is being developed by staff on temporary contracts (PhD students, etc.).
 - ...the lack of on-the-job training for better software development practices.
 - ...the lack of paid positions for research software engineers.
- Providing a set of communications tools to facilitate exchange between research software engineers affiliated with different PSI Centers.
- Providing a community-powered list of PSI-specific RSE resources on the new rse.psi.ch website to augment the intranet, local wiki pages and other documentation.
- Contributing to Swiss-wide efforts coordinated by the Swiss RSE Association rse.swiss.

References and Further Reading

Cohen et al. "The four pillars of research software engineering."
IEEE Software (2020). doi:[10.1109/MS.2020.2973362](https://doi.org/10.1109/MS.2020.2973362)

Barker et al. "Introducing the FAIR Principles for research software".
Scientific Data (2022). doi:[10.1038/s41597-022-01710-x](https://doi.org/10.1038/s41597-022-01710-x)

Jensen and Katz. "Awareness of FAIR and FAIR4RS among international research software funders."
Scientific Data (2025) doi:[10.1038/s41597-025-04820-4](https://doi.org/10.1038/s41597-025-04820-4)

Chue Hong et al. "FAIR principles for research software (FAIR4RS principles)."
Zenodo (2022). doi:[10.15497/RDA00068](https://doi.org/10.15497/RDA00068)

Meloca et al. "Understanding the usage, impact, and adoption of non-OSI approved licenses."
Proceedings of the 15th international conference on mining software repositories (2018). doi:[10.1145/3196398.3196427](https://doi.org/10.1145/3196398.3196427)

Research Software Alliance. "Amsterdam Declaration on Funding Research Software Sustainability (1.1)".
Zenodo (2022). doi:[10.5281/zenodo.13735888](https://doi.org/10.5281/zenodo.13735888)

See also the websites of:

