The State of Sustainable Research Software: Learning from the Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE5.1)

Daniel S. Katz[*], Stephan Druskat[†], Robert Haines[‡], Caroline Jay[§], Alexander Struck[¶]

December 4, 2018

Abstract

This paper uses the accepted submissions from the Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE5.1) held in Manchester, UK in September 2017 and the speed blogs written during the event to examine the state of research software. It presents a schematic of the space, then examines coverage in terms of topics, actors, actees, and themes by both the submissions and the blogs.

[*]National Center for Supercomputing Applications (NCSA) & Department of Computer Science & Department of Electrical and Computer Engineering & School of Information Sciences (iSchool), University of Illinois, Urbana–Champaign, IL, USA; d.katz@ieee.org; ORCID: 0000-0001-5934-7525

[†]Department of German Studies and Linguistics, Humboldt-Universität zu Berlin, Berlin, Germany; stephan.druskat@hu-berlin.de; ORCID: 0000-0003-4925-7248

[‡]Research IT, University of Manchester, Manchester, UK; robert.haines@manchester.ac.uk; ORCID: 0000-0002-9538-7919

[§]School of Computer Science, University of Manchester, Manchester, UK; caroline.jay@manchester.ac.uk; ORCID: 0000-0002-6080-1382

[¶]Cluster of Excellence Image Knowledge Gestaltung at Humboldt-Universität zu Berlin; Germany; Alexander.Struck@hu-berlin.de; ORCID: 0000-0002-1173-9228

1 Introduction

In September 2017, 37 people interested in sustainable research software came together at the Working towards Sustainable Software for Science: Practice and Experiences (WSSSPE5.1, wssspe.researchcomputing.org.uk/wssspe5-1/) meeting in Manchester, UK. WSSSPE5.1 immediately preceded the Second Research Software Engineers (RSE) Conference, so that RSE attendees could also attend WSSSPE5.1.

WSSSPE is an international community-driven organization that promotes sustainable research software by addressing challenges related to the full lifecycle of research software through shared learning and community action. It envisions a world where research software is accessible, robust, sustained, and recognized as a scholarly research product critical to the advancement of knowledge, learning, and discovery.

WSSSPE promotes sustainable research software by positively impacting:

- Principles and Best Practices. Promoting best practices in sustainable software
- Careers. Developing and supporting career paths in research software development and engineering
- Learning. Engaging in activities to promote peer learning and interaction
- Credit. Ensuring recognition of research software as an intellectual contribution equal to other research products

WSSSPE defines *Sustainable software* as software that has the capacity to endure such that it will continue to be available in the future, on new platforms, meeting new needs. The *research software lifecycle* includes: acquiring and assembling resources (including funding and people) into teams and communities, developing software, using software, recognizing contributions to and of software, and maintaining software.

Six previous WSSSPE events[1] [1, 2, 3, 4] included group discussions about problems and potential solutions in the sustainable research software space. WSSSPE5.1 used the speed blog methodology to generate eight reports on

[1]The first WSSSPE workshop was named "Working towards Sustainable Software for Science: Practice and Experiences," which remains the meaning of the WSSSPE group, but the workshops after that were named "Workshop on Sustainable Software for Science: Practice and Experiences." Together these reflect that WSSSPE is both a community and a set of workshops.

different views of this space. The blogs were published by the UK Software Sustainability Institute (SSI) on its website.

The remainder of this paper includes a set of presentations from the accepted papers and lighting talks (§2), outputs from the speed-blogging groups (§3), and thematic analysis of the resulting blogs (§4), before concluding (§5).

2 Presentations

Submissions to WSSSPE5.1 comprised six papers and eight lightning talks. Of these, four papers [5, 6, 7, 8] and seven lightning talks [9, 10, 11, 12, 13, 14, 15] were accepted. All papers and lightning talks were published as a figshare collection [16]. Slides for the given talks have been published on the WSSSPE5.1 website (wssspe.researchcomputing.org.uk/wssspe5-1/wssspe5-1-agenda/).

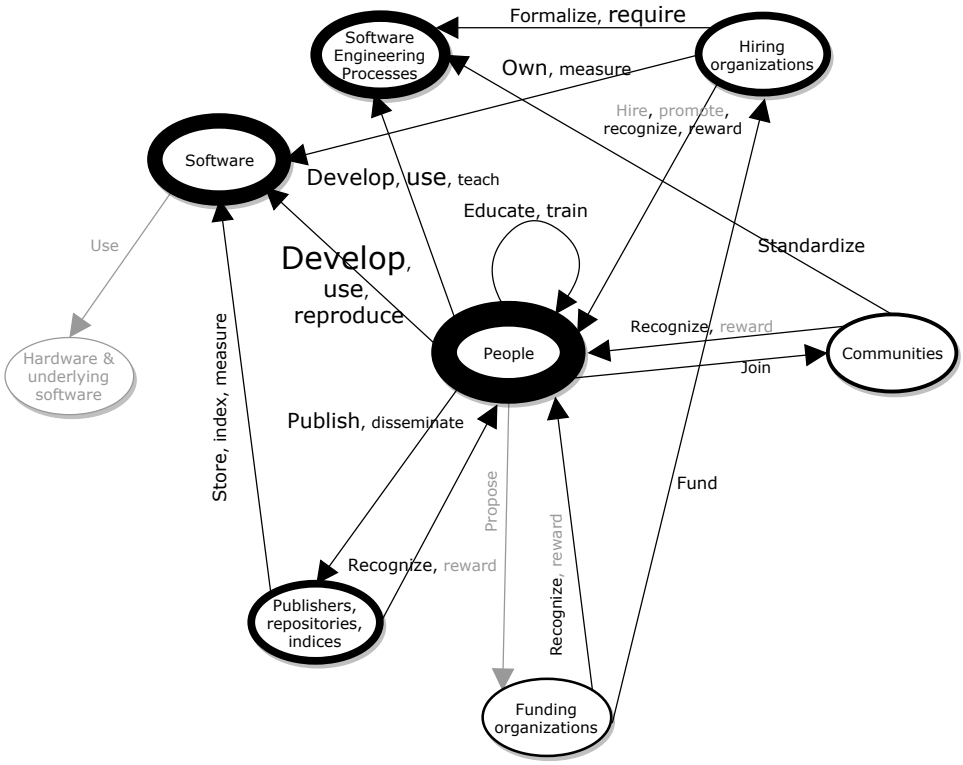

Figure 1: Research software sustainability space schematic; topic weights based on WSSSPE5.1 presentations.

The "research software sustainability space" can be described as a set of activities which impact the sustainability of research software in different ways and on different levels, e.g., how a research software is developed by developers, how it is funded, how and where it is published, etc. These activities consist of "actions", which are undertaken by "agents", where an "actor" acts on an "actee". Activities in the space can thus be modeled as a directed graph, and represented in respective schematics. In this section, we aim to evaluate which subset of activities has been represented in the presentations given at WSSSPE5.1.

Figure 1 shows a version of a schematic of activities within the research software sustainability space as introduced by Katz [17]. The original schematic has been a result of introspective work based on experience rather than quantitative research. Author Druskat has remodeled the original schematic manually, using [18] run on draw.io, to produce Figure 1. In order to adapt it for purposes of workshop evaluation, both node outlines and edge labels have been weighted to show the distribution of recorded agents and actions across workshop presentations [19]. This work was done based on the presentation abstracts [16] rather than the actual presentations to ensure traceability.

To calculate the applicable weights, the schematic was resolved into unique activities. A unique activity is a single edge label verb (the action) going from a node (the actor) to another node (the actee). The presentations were then coded, recording for each presentation whether or not an activity was present. Based on the total of occurrences of an activity across presentations (x) and the range of occurrence totals of all activities across presentations (r), the font size in point (pt) for the edge labels representing single actions has been normalized to target scaling range $t = 12 - 24$, yielding the scaled font size x':

$$x' = \frac{x - r_{min}}{r_{max} - r_{min}} \cdot (t_{min} - t_{min}) + t_{min}$$

Similarly, the total of occurrences of an agent across presentations, either as actor or actee of an activity, has been normalized to target scaling range $t = 2 - 20$, yielding the scaled width of the node outline in pt.

Figure 2 shows the distribution of actions over all presentations, based on [19], where an action is a labeled edge from a node in the sustainability schematic (Figure 1) to another node. The source node represents the actor, and the target node the actee of the respective action. The weights and action distribution show a clear overall primary focus of presentations on software development (*People develop Software*). More generally, the people involved

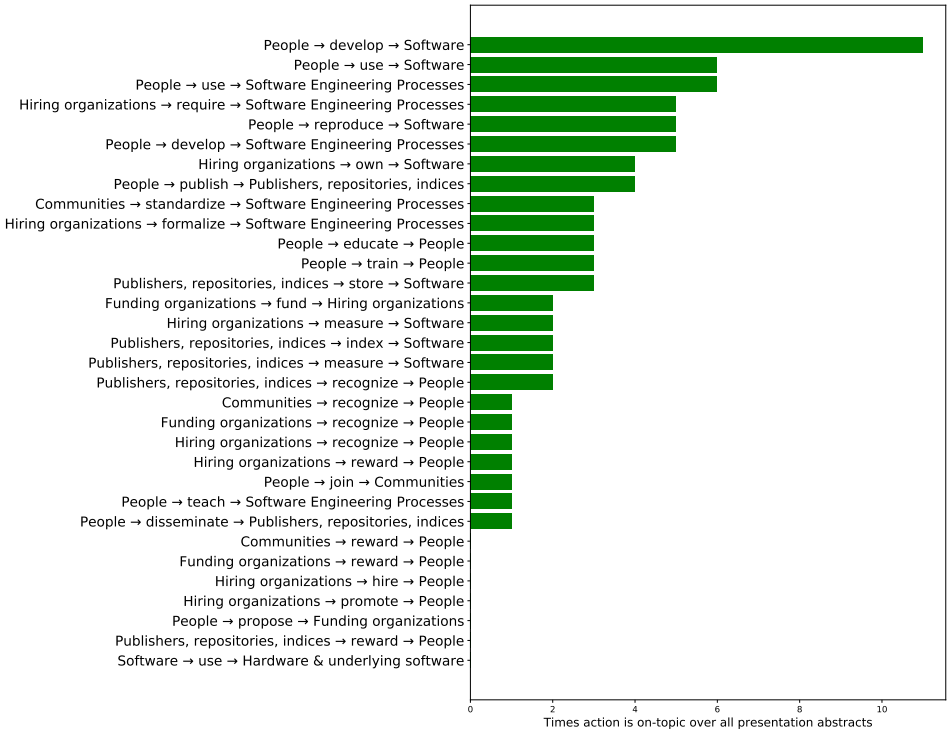

Figure 2: Distribution of activity coverage over presentations.

in research software take prominent focus as well as software engineering principles, and research software itself. However, across all presentations most actors, actees, and actions within the space have been the topic of a presentation as actor or actee, with the exception of hardware and underlying software. Future workshops could take care to address this topic specifically in their calls for submissions in order to close gaps in research, discussion and progress.

Taken for itself, these figures only represent a snapshot of activities around research software sustainability. In order to represent the continuum of efforts and make quantitatively informed statements about both general progress, and the specific insights and outcomes from the particular workshop reported on here, a larger corpus of workshop products (abstracts, presentations, blogs, etc.) would have to be analyzed. While such a larger-scaled analysis is out of scope for this report, work has started within the WSSSPE community to create an ontology of activities in the research software sustainability space [20, 21], which will enable analyses of this kind in future work.

Table 1 shows the distribution of combined actor and actee reference from the research software sustainability space over workshop presentations.

Presentation	Communities	Funding organizations	Hardware & underlying software	Hiring organizations	People	Publishers, repositories, indices	Software	Software engineering processes
[5]					•		•	
[6]	•			•	•		•	•
[7]					•		•	•
[8]				•	•		•	•
[9]		•		•	•		•	
[10]		•		•	•		•	•
[11]				•	•		•	•
[12]				•		•	•	
[13]					•	•	•	•
[14]	•			•	•	•	•	•
[15]	•	•		•	•	•	•	•

Table 1: Distribution of topics from the research software sustainability space over workshop presentations for actors and actees (combined).

3 Speed blogs

The Software Sustainability Institute (SSI) discusses speed blogging at www.software.ac.uk/term/speed-blogging. The goal of speed blogging is to preserve as much content and context as possible from working groups, and to publish results in an easily digestible form, usually blog posts. Half the available working group time at WSSSPE5.1 was allocated to discussion, and the other half to writing the blogs. The rest of this section summarizes the WSSSPE5.1 blog posts, which are accessible at wssspe.researchcomputing.org.uk/wssspe5-1/. Indented italicized text indicates quotations from the blogs.

- The Research Software Project Manager (www.software.ac.uk/blog/2017-12-04-research-software-project-manager)

  > *For many, the role of research software project manager (RSPM) may be an accidental calling. The career path for this role isn't well-established, and research software development in academia may itself be something of a haphazard,*

> *nigh-accidental byproduct of conducting domain research. Individuals approaching this role may have little to no wider industry experience, instead approaching the project manager role from research or research software engineering.*

The authors present the challenges for the RSPM role, noting that it is inherently different from similar positions in the industry. The focus of work is on enabling research as an iterative process, prone to many changes along the way. While many academics excel at winning grants and pushing the scientific frontiers, only a few are trained in software project management, and those who are may focus on technical aspects, discarding parts of the overall picture. The blogpost raises a number of questions about this situation, and suggests answers to a few of them, including strengthening the role of the RSPM and considering positions dedicated to project management. Such a delegation of tasks would smooth the process of reporting to administration and free up necessary resources for actual research.

- Looking for software use in research
(www.software.ac.uk/blog/2017-12-05-looking-software-use-research)

> *Nowadays, software is used in most research. But how the software is created, used, and what it depends on are not well understood questions. The importance of such knowledge varies based on the motivation of the reader. On one side, we could be interested in the impact of the software, how many times it has been used and by who. This type of analysis could come, for example, from funding bodies and organisations to reward the creation of something and help its sustainability, from institutions who hire people behind that software, or from the software authors to get an understanding of the needs of their users or simply to get credit for their work. Another motivation may be trying to understand the research being carried out with a particular software or set of tools either for purely academic purposes (e.g., by historians and scholars of science) or with a commercial perspective (such as by intellectual property teams from universities for the monetisation of the software).*

The blog post reports on methods for finding software. Today, researchers usually consult search engines and programming language package archives. In the future, Current Research Information Systems

(CRIS) or repositories maintained by funders may hold information about software developed during research. Software as a research output is increasingly registered with DOIs [22] but identifying software in written text still has its challenges [23]. Repositories such as Bioconductor offer a domain-specific overview of available tools. General purpose repositories, such as GitHub.com, track usage as forks on their platforms, whereas services like Depsy.org and libraries.io provide aggregated software usage information from many sources. Software, considered by some to be a subset of research data, could also be identified via DOIs, minted by services like DataCite.org that point to published data and software.

- Towards Reproducibility in Research Software (www.software.ac.uk/blog/2017-12-06-towards-reproducibility-research-software)

  > *Ensuring reproducibility of research has been identified as one of the challenges in scientific research. While reproducibility of results is a concern in all fields of science, the emphasis of this group is in the area of computer software reuse and the reproduction of results. The availability of complete descriptions, ideally including program source code, documentation and archives of all necessary components and input datasets would be a major step to resolving research reproducibility concerns.*

  A short review of relevant services and platforms is provided by the authors. It includes concepts from software engineering, e.g., version control, being introduced in programmes such as Software Carpentry. Platforms and tools, such as Overleaf or Jupyter notebooks, ease collaborative review of research and reproducibility of code respectively. While the uptake of such best practices is slow and the lack of documentation sometimes hinders reproducibility, we may see more of them in the future, especially if incentives are set right.

- Why research software engineers should have permanent contracts (www.software.ac.uk/blog/2017-12-11-why-research-software-engineers-should-have-permanent-contracts)

  At present, only a few research institutions employ research software engineers and make their resources available to the whole organization. This blog post discusses some success stories motivating long term con-

tracts for RSEs. In general "better software [leads to] better research" (SSI).

> *Higher quality output is on every university's wish list, as it leads to a potential increase in QR funding (www.hefce.ac.uk/rsrch/funding/mainstream/), while reducing the reputational risk associated with substandard research practices. Reproducibility is notoriously difficult to achieve and RSEs are an essential part of enabling this [...] leading to higher citations and greater research impact.*

Skilled RSEs are sought after and organizations get what they pay for. The "stability and the potential for some form of career progression" that permanent contracts offer should be a consideration when hiring such staff. And the costs do not necessarily reduce the university's baseline funds as research projects are encouraged to "cost in" consulting by centrally pooled RSEs. Long-term benefits for organizations are the spreading of best practices in software development and an institutional memory provided by the central RSE group who have worked in many projects.

> *An RSE team that is permanently employed can be truly agile. Recruiting an expert for a short period of time in an academic institution is virtually impossible. As long as we rely on fixed-term contracts for RSEs, a lot of important work will fail to be done, and funds will not be spent as effectively as they could be.*

- A standard format for CITATION files (www.software.ac.uk/blog/2017-12-12-standard-format-citation-files)

> *The citation of research software has a number of purposes, most importantly attribution and credit, but also the provision of impact metrics for funding proposals, job interviews, etc. Stringent software citation practices [...] therefore include the citation of a software version itself, rather than a paper about the software. Direct software citation also enables reproducibility of research results as the exact version can be retrieved from the citation.*

While a diverse range of citation recommendations exist for software projects, many projects still proceed without a proper citation. In order

to unify the various existing procedures, the authors proposed the inclusion of a CITATION file in a software repository, holding standardized information, e.g., ORCID to enable linking authors with their respective ID from orcid.org. It shall be easy to read and write in order to enable fast and straightforward creation of citation information. For this purpose, the site research-software.org/citation/ was created and is actively maintained to cover all aspects of research software citation. Widespread use of the Citation File Format (CFF) may enable transitive credit information provision and better impact metrics next to reusability by other actors.

- Overcoming barriers to adopting software best practices in research (www.software.ac.uk/blog/2017-12-07-overcoming-barriers-adopting-software-best-practices-research)

  Research careers create a wide spectrum of skill levels (compared to corporate environments). Training a group of researchers on a new topic is challenging, given the lack of a peer group at some research frontiers or missing institutional support for training and the significant upfront cost involved in learning e. g., best development practices. Researchers who mostly work on code in isolation rarely have an opportunity to have code reviews (with a notable exception as described in www.software.ac.uk/blog/2018-05-18-code-review-academia), need to learn on their own, and are often not recognized for writing code. Paper publication in a "high-impact peer-reviewed journal" still pushes careers. A way to overcome barriers is to introduce, e.g., "industry standards" in a manner adapted to research environments. The blog post discusses how the "SCRUM" approach may not work, but "agile" or a "maturity model" approach may better fit the current research practices.

  > *As a general concept: start small and then go as far as necessary. Reaching for the perfect software development approach is intimidating and overwhelming, and it is not the task of a researcher nor necessary for most research projects. A maturity model can help researchers identify where they are and where they should be [. . .]. Restricting the use of tech jargon to a minimum and offering explanations where necessary can help, too.*

- Encouraging good software development practice in research teams (www.software.ac.uk/blog/2017-12-13-encouraging-good-software-development-practice-research-teams)

"Training in good software development skills is vital for the uptake and maintenance of good practice in a research community." This requires that resources, tools, and sometimes management support the availability of, and that the trainee be motivated. The latter could be achieved when the "connection between good practice and increased publication rate and quality, as well as funding availability" are demonstrated. A "consistent approach to software development [...] can benefit collaboration and the longevity of projects." Success story from the DLR (www.dlr.de) and the EMBL (www.embl.de) are reported in the blog post, which goes into detail on how to build a community and keep it alive. The authors propose the concept of "inner source," where open source principles are utilized collaboratively inside an organization to develop "common scientific frameworks." GitLab is praised as a beneficial tool that eases adoption of good practices. The authors recognize that rollout of training, tools, and services should happen iteratively and is always highly dependent on the particular environment. Therefore they are interested to learn from others and encourage feedback via e-mail.

- Overcoming Entry Barriers to Motivate Better Practice in Research Software Engineering (www.software.ac.uk/blog/2017-12-14-overcoming-entry-barriers-motivate-better-practice-research-software-engineering)

  > *What can be termed as "coding" is a subset of wider software engineering practices such as version control, continuous integration and good software design. Coding is prevalent in academia but practices that allow sustainable software to be produced are frequently overlooked. Motivating the uptake of the approaches, methods and tools, and highlighting the benefit they deliver, by engaging with researchers who develop software is the first step in spreading best practice in our community.*

  The authors point out the benefit of using online systems such as GitLab to reduce entry barriers and motivating the use of, e.g., version control and continuous integration (CI). Other software engineering principles such as pair-programming and code review are also encouraged early in (graduate) students training in order to demonstrate the benefits for future use and bridging gaps between disciplines. If these research software management practices become a requirement in grants application and reporting, widespread deployment is inevitable. Increasing recognition of software as a valid research output could provide further motivation, along with better reproducibility and reduced duplication of effort.

4 Analyzing the speed blogs

The speed blog topics were determined using an unconference format—where participants chose what to cover as a group—and therefore provided a snapshot of issues that were particularly timely and relevant to the WSSSPE community. Participants were free to choose which speed blogging group to join, depending on their own particular interests and goals. In this section, we treat the speed blogs as qualitative data, which we systematically analyse to determine the prevalence of particular themes.

4.1 Method

We used a hybrid thematic/framework analytic approach, where we used the schematic of the research software sustainability space shown in Figure 1 as a starting point for our analysis, and then refined this as we familiarized ourselves with the data. The schematic nodes formed the categories: *funders* ('funding organizations' in the schematic); *employers* ('hiring organizations'); *publishers, repositories, indices*; *research software* ('software'); *software engineering processes*; *communities*. The edges provided the categories: *reward & recognition* ('recognize, reward'); *training*; *standardization* ('standardize'); *reproducibility* ('reproduce'). Based on a bottom-up analysis of the data, we broke the schematic 'people' node down further into *users*, *research software engineers* and *researchers*, and added a further category of *software infrastructure*.

Authors Jay and Haines coded the blogposts independently, recording for each blogpost whether or not the theme was present. This process resulted in agreement of 79%. Disagreements were then resolved via discussion. The original data set (blogposts and individual and joint coding scores) are available for further analysis [24].

4.2 Results

All of the eight blogs mentioned research software, and researchers (i.e., domain specialists rather than RSEs). The blogs also all mentioned reward and recognition, either for software itself, or for people writing software. The prevalence of all of the categories across the blog posts can be seen in Figure 3. During the discussion of the speed blog topics it was decided that there seemed to be sufficient writing already on citation and credit, so while this topic fit under the WSSSPE umbrella, it was not suggested as a topic for speed blogs. The fact that it was mentioned in all the blogs indicates

that it is still an important theme for the community.

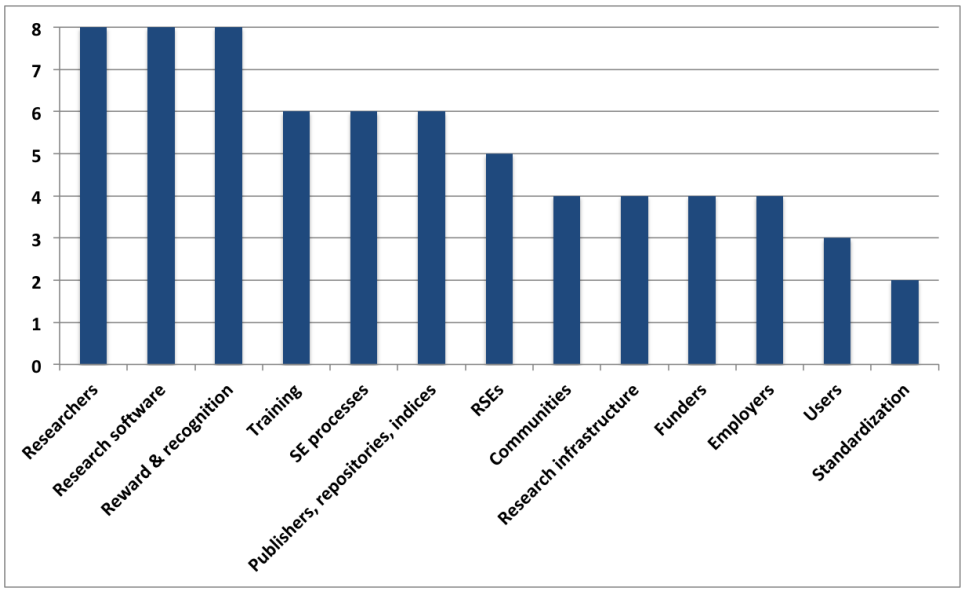

Figure 3: The number of speed blogs mentioning each of the themes.

5 Conclusions

In summary, the community recognizes the need for improving software engineering practices and has been starting to take action around this. A set of initial success stories have been reported from a handful of organizations. Recognition for work, including via citations, remains a topic of interest, with less progress that is needed. Based on the presentations and working groups at WSSSPE5.1, we have presented a set of topics and mapped the presentations and speed blogs onto these topics. The presentations and speed blogs cover most of the topics and do not have subjects that are not in the list, indicating that the topics (or themes, a more fine-grained mapping of the space as discussed in Section 4) are a good representation of the space.

While the existing set of topics may serve as a broad overview of issues in the research software sustainability space, we cannot yet use them to make broader, informed statements about how the space itself and the activities taking place within it have developed, or whether – and which – activities have helped to solve issues, and whether there are gaps in activities that the community should aim to fill.

Overall, as we have stated previously [25], we have learned from the first four years of WSSSPE that it is relatively easy to get motivated people to attend a meeting and productively spend their time there both doing work and planning more work, but it is very hard to get that additional work after the meeting to take place. Given this, we have turned WSSSPE meetings (including this one and another in 2017, and one in 2018) into gathering places to discuss scientific software sustainability, and for groups that are already in place or that can be composed of related funded activities to meet.

Acknowledgments

S. Druskat would like to acknowledge funding assistance from the Software Sustainability Institute. The Software Sustainability Institute is supported by the EPSRC, BBSRC and ESRC Grant EP/N006410/1.

References

[1] Katz DS, Allen G, Chue Hong N, Parashar M, Proctor D. First Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE): Submission and Peer-Review Process, and Results. arXiv; 2013. 1311.3523. Available from: http://arxiv.org/abs/1311.3523.

[2] Katz DS, Choi SCT, Lapp H, Maheshwari K, Löffler F, Turk M, et al. Summary of the First Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE1). Journal of Open Research Software. 2014;2(1). Available from: https://doi.org/10.5334/jors.an.

[3] Katz DS, Allen G, Chue Hong N, Cranston K, Parashar M, Proctor D, et al. Second Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE2): Submission, Peer-Review and Sorting Process, and Results. arXiv; 2014. 1411.3464. Available from: http://arxiv.org/abs/1411.3464.

[4] Katz DS, Choi ST, Wilkins-Diehr N, Chue Hong N, Venters CC, Howison J, et al. Report on the Second Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE2). Journal of Open Research Software. 2016;4(1):e7. Available from: https://doi.org/10.5334/jors.85.

[5] Nangia U, Katz DS. Track 1 Paper: Surveying the U.S. National Postdoctoral Association Regarding Software Use and Training in Research. In: Chue Hong N, Druskat S, Haines R, Jay C, Katz DS, Sufi S, editors. Proceedings of the Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE5.1). figshare; 2017. https://doi.org/10.6084/m9.figshare.5328442.v3.

[6] Haupt C, Schlauch T. Track 1 Paper: The Software Engineering Community at DLR — How We Got Where We Are. In: Chue Hong N, Druskat S, Haines R, Jay C, Katz DS, Sufi S, editors. Proceedings of the Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE5.1). figshare; 2017. https://doi.org/10.6084/m9.figshare.5331703.v2.

[7] Queiroz F, Silva R, Miller J, Brockhauser S, Fangohr H. Track 1 Paper: Good Usability Practices in Scientific Software Development. In: Chue Hong N, Druskat S, Haines R, Jay C, Katz DS, Sufi S, editors. Proceedings of the Workshop on Sustainable Software

for Science: Practice and Experiences (WSSSPE5.1). figshare; 2017. https://doi.org/10.6084/m9.figshare.5331814.v3.

[8] Mulholland D, Alencar P, Cowan D. Track 2 Paper: The Future of Metadata-Oriented Testing of Research Software: Automated Generation of Test Regimes and Other Benefits. In: Chue Hong N, Druskat S, Haines R, Jay C, Katz DS, Sufi S, editors. Proceedings of the Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE5.1). figshare; 2017. https://doi.org/10.6084/m9.figshare.5334091.v1.

[9] Silva R. Track 1 Lightning Talk: Research Software in Brazil. In: Chue Hong N, Druskat S, Haines R, Jay C, Katz DS, Sufi S, editors. Proceedings of the Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE5.1). figshare; 2017. https://doi.org/10.6084/m9.figshare.5328043.v1.

[10] Struck A. Track 1 Lightning Talk: How Red Tape and Other Obstacles Are Holding Us Back. In: Chue Hong N, Druskat S, Haines R, Jay C, Katz DS, Sufi S, editors. Proceedings of the Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE5.1). figshare; 2017. https://doi.org/10.6084/m9.figshare.5350501.v2.

[11] Washbrook A. Track 1 Lightning Talk: Continuous Software Quality Analysis for the ATLAS Experiment at CERN. In: Chue Hong N, Druskat S, Haines R, Jay C, Katz DS, Sufi S, editors. Proceedings of the Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE5.1). figshare; 2017. https://doi.org/10.6084/m9.figshare.5348830.v2.

[12] Dasler R. Track 1 Lightning Talk: CERN Analysis Preservation - Contextualising Analyses through Data and Software Preservation. In: Chue Hong N, Druskat S, Haines R, Jay C, Katz DS, Sufi S, editors. Proceedings of the Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE5.1). figshare; 2017. https://doi.org/10.6084/m9.figshare.5330812.v2.

[13] Alhozaimy S, Haines R, Jay C. Track 1 Lightning Talk: Forking as a Tool for Software Sustainability —An Empirical Study. In: Chue Hong N, Druskat S, Haines R, Jay C, Katz DS, Sufi S, editors. Proceedings of the Workshop on Sustainable Software for Science: Practice and

Experiences (WSSSPE5.1). figshare; 2017. https://doi.org/10.6084/m9.figshare.5328796.v1.

[14] Maassen J, Drost N, van Hage W, van Nieuwpoort R. Track 2 Lightning Talk: Software Development Best Practices at the Netherlands eScience Center. In: Hong NC, Druskat S, Haines R, Jay C, Katz DS, Sufi S, editors. Proceedings of the Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE5.1). figshare; 2017. https://doi.org/10.6084/m9.figshare.5327587.v2.

[15] Druskat S. Track 2 Lightning Talk: Should CITATION Files Be Standardized? In: Chue Hong N, Druskat S, Haines R, Jay C, Katz DS, Sufi S, editors. Proceedings of the Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE5.1). figshare; 2017. http://doi.org/10.6084/m9.figshare.3827058.

[16] Chue Hong N, Druskat S, Haines R, Jay C, Katz DS, Sufi S, editors. Proceedings of the Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE5.1). University of Manchester, Manchester, UK: figshare; 2017. Available from: https://doi.org/10.6084/m9.figshare.c.3869782.

[17] Katz DS. Research Software Sustainability: WSSSPE & URSSI; 2018. Available from: https://doi.org/10.6084/m9.figshare.6081248.v1.

[18] The JGraph drawio authors. draw.io (Version v8.7.10); 2018. Available from: https://github.com/jgraph/drawio/releases/tag/v8.7.10.

[19] Druskat S. WSSSPE5.1 presentations: Research software sustainability activity analysis (Version 0.2.0); 2018. Available from: https://doi.org/10.5281/zenodo.1291506.

[20] Druskat S, Katz DS. Mapping the research software sustainability space; 2018. Available from: https://arxiv.org/abs/1807.01772.

[21] Druskat S, Katz DS. Mapping the Research Software Sustainability Space. In: 2018 IEEE 14th International Conference on eScience (eScience). IEEE Computer Society; forthcoming. p. 25–30. Available from: https://doi.org/10.1109/eScience.2018.00014.

[22] Fenner M, Katz DS, Nielsen LH, Smith A. DOI Registrations for Software. DataCite; 2018. Available from: https://blog.datacite.org/doi-registrations-software/.

[23] Li K, Yan E, Feng Y. How is R cited in research outputs? Structure, impacts, and citation standard. Journal of Informetrics. 2017 nov;11(4):989–1002. Available from: https://doi.org/10.1016/j.joi.2017.08.003.

[24] Jay C, Haines R. WSSSPE 5.1 - Data for speed blog analysis; 2018. Available from: https://doi.org/10.5281/zenodo.1305091.

[25] Katz DS, Niemeyer KE, Gesing S, Hwang L, Bangerth W, Hettrick S, et al. Fourth Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE4). Journal of Open Research Software. 2018;6(1):10. Available from: https://doi.org/10.5334/jors.184.