Software as a first-class citizen in research

Leyla Garcia[1*][0000-0003-3986-0510], Michelle Barker[2][0000-0002-3623-172X], Neil Chue Hong[3][0000-0002-8876-7606], Fotis Psomopoulos[4][0000-0002-0222-4273], Jennifer Harrow[5][0000-0003-0338-3070], Daniel S. Katz[6][0000-0001-5934-7525], Mateusz Kuzak[7][0000-0003-0087-6021], Paula Martínez[8][0000-0002-8990-1985], Allegra Via[9][0000-0002-3398-5462]

[1] ZB MED Information Centre for Life Sciences, Cologne, Germany
[2] Research Software Alliance, https://www.researchsoft.org/
3 Software Sustainability Institute - University of Edinburgh, Edinburgh, UK.
[4] Institute of Applied Biosciences, Centre for Research and Technology Hellas, Thessaloniki, Greece
[5] ELIXIR Hub, Hinxton, UK
[6] University of Illinois Urbana-Champaign, Urbana, IL, USA
[7] Netherlands eScience Center, Amsterdam, Netherlands
[8] The Centre for Advanced Imaging - The University of Queensland, Queensland, Australia
[9] National Research Council, Roma, Italy
* ljgarcia@zbmed.de

Abstract. In recent years the importance of software in research has become increasingly recognized by the research community. This journey still has a long way to go. Research data is currently backed by a variety of efforts to implement and make FAIR principles a reality, complemented by Data Management Plans. Both FAIR data principles and management plans offer elements that could be useful for research software but none of them can be directly applied; in both cases there is a need for adaptation and then adoption. In this position paper we discuss current efforts around FAIR for research software that will also support the advancement of Software Management Plans. In turn, use of SMPs encourages researchers to make their datasets FAIR.

Keywords: Research software, Management Plan, FAIR

1 Background

One of the major challenges in data-driven research is facilitating knowledge discovery by assisting humans and machines in their discovery of, access to, and integration and analysis of data and their associated research objects, e.g., algorithms, software, and workflows. Both publications, which remain the most common means to disseminate research results, and data are recognized as important elements of research. Data is used as input or created, then analyzed, integrated, and transformed to become an output, contributing to obtaining new insights and therefore advancing science.

The FAIR data principles [1] strongly contribute to addressing this challenge with regard to research data, and the principles, at a high level, are intended to apply to all

research objects; both those used in research and those being produced as research outcomes. The FAIR data principles can, and should, be complemented with Data Management Plans (DMPs) as they both contribute to improving (meta)data quality. DMPs are documents describing methods, techniques, and policies regarding how data is managed from beginning to end during a research project [2].

In recent years, the role of software has slowly become more widely recognized as essential in research. While software has been essential in some research fields for many decades (e.g.. climate and weather for 70+ years, bioinformatics for 40+ years), this was not well appreciated by those who do not directly implement or use the software. FAIR principles and management support are not yet as advanced for software as for data. Many of the high-level FAIR data principles can be directly applied to research software by treating software and data as similar digital research objects. However, specific characteristics of software — such as its executability, composite nature, and continuous evolution and versioning — make it necessary to revise and extend the original data principles. Some elements from DMPs also apply to software, particularly those related to purpose, provenance, documentation, findability and accessibility. However, similarly to the FAIR principles, there are fundamental differences between data and software that must be recognized and addressed by Software Management Plans (SMPs) — those already mentioned plus some others such as testing —.

In this position paper we discuss the current status of FAIR principles applied to research software and introduce some basic elements for SMPs. We explain how development of each of these supports the other, and invite the reader to get involved in current initiatives around these two efforts, FAIR for research software and SMPs, that are taking software to its next stage of recognition in research. Our goal is to reach the stage where software is fully recognized and integrated as a first-class citizen in research.

2 FAIR principles for Research Software

The FAIR principles are meant to provide guidance around findability, accessibility, interoperability and reusability. However, they do not provide implementation details. Initial efforts on implementation have mainly revolved around data, e.g., the Research Data Alliance (RDA)[1] working group on a FAIR Data Maturity Model [3]. In the past two years, the research software community has been active in finding ways to make FAIR principles a reality for research software, only to find out that the principles as initially stated cannot be directly applied to software. There is a need for some adaptation, rephrasing, and extension [4].

The subject of FAIR for research software has been discussed in multiple scientific events and has led already to some publications on the field [4,5], see Fig. 1. Additional efforts focus on software citation [6,7] software benchmarking [8] and software metadata [9]. With the aim of unifying efforts across multiple disciplines and leading the research software community in the crucial step of agreeing on the

[1] https://www.rd-alliance.org/

application of the FAIR principles to research software by mid-2021, the FAIR for Research Software (FAIR4RS) working group was formed in April 2020 [10]. The FAIR4RS WG is jointly convened as a Research Software Alliance (ReSA)[2] Taskforce, an RDA Working Group, and a FORCE11[3] Working Group, in recognition of the importance of this work across the research sector. When drafting of the principles completes in mid-2021, they will need to be applied to a range of other areas, including SMPs, metrics, incentives, skills and FAIR services that provide persistent identifiers, metadata specifications, stewardship and repositories, and actionable policies.

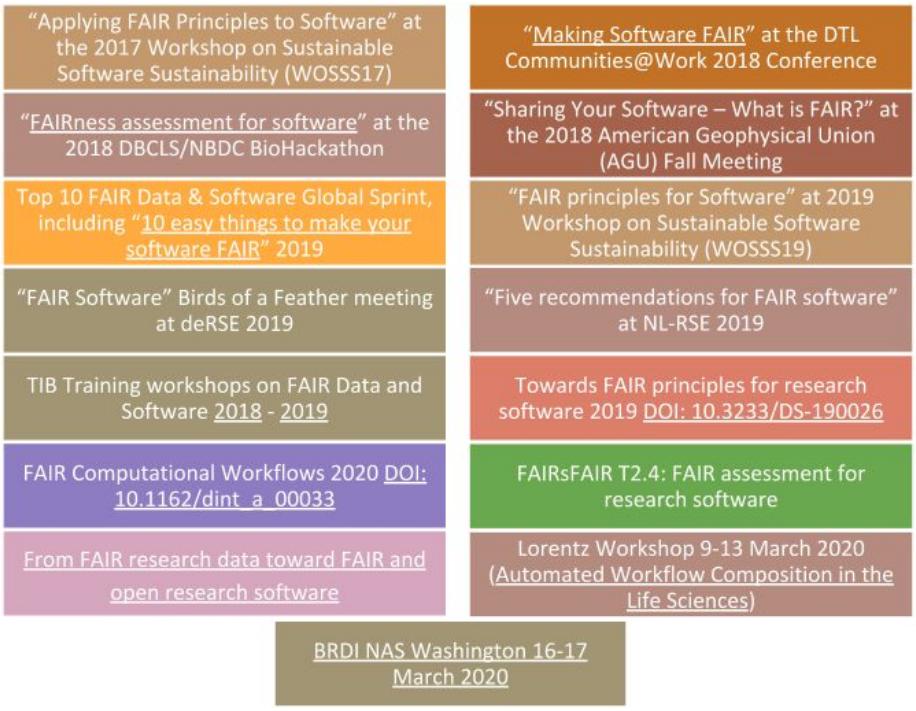

Fig. 1. Some efforts including events and publications around FAIR 4 research software. Adapted from [11].

3 Software versus Data

There are many inherent differences between software and data, and additionally, there are differences between how they are created, maintained, and used in the scholarly system. Software is data, but it is not *just* data. While "data" in computing and information science can refer to anything that can be processed by a computer,

[2] https://www.researchsoft.org/
[3] https://www.force11.org/

software is a special kind of data that can be a creative, executable tool that operates on data. Specific differences, as now being developed in the FAIR4RS activity based on [6], include:

- Software is executable, data is not.
- Data provides evidence, software provides a tool.
- Software is a creative work, scientific data are facts or observations. Related to this, software licenses are different than data licenses, and in many countries, software is subject to copyright protection while data is not.
- Software suffers from software collapse (software is typically built to use other software, leading to complex dependencies on this software, and these dependent software packages also frequently change, leading the software to stop working, or collapse)), where data and software both suffer from bit rot (they cannot be read due to changes in media and storage).
- Software (and scientific software especially) is sometimes highly optimised for the hardware on which it runs, making it far more dependent on changes to that hardware, while data is more commonly expressed in a form abstracted away from these concerns.
- The lifetime of software is generally not as long as that of data.
- Software, over its lifetime, is typically subject to many changes whereas data often is not.
- Much software is shared while it is being developed, and it is shared via social coding platforms such as GitHub, while much data is not shared until it is published in a preservation/archival data repository.

4 Software Management Plans

SMPs, similar to DMPs, are documents describing elements that should be considered during the lifecycle of research software, from beginning to end. SMPs help researchers and research software developers understand, at a basic and practical level, what processes, resources, and infrastructures are required and how they may be used to achieve development goals [12]. As research software encompasses code and solutions from scripts to production level software, it is better to initially focus on the minimum desirable elements. As it happens with the implementation of FAIR principles, there is also a need for defining maturity models that can help researchers and funders focus on the aspects that are most relevant in a given research project. It is important to note that a SMP is not the same as a Software Project Management as the latter is broader, more complex, and oriented more to Software Engineer than research.

SMPs in research have existed since 2014 [13] but have not yet been widely adopted, although some DMPs now include software in their focus. However, given the differences between software and data, we argue that SMPs should complement DMPs and having them separated allows a better focus on two equally important research objects, i.e., data and software. Increased usage of SMPs would also

encourage researchers to make their software FAIR. Furthermore, a growth in funders requiring SMPs from the start of their project would further drive behavioural changes, and signify increased recognition by funders of software outputs.

The Software Best Practices Task Force at the ELIXIR Europe Tools Platform[4] is defining a SMP specifically for the Life Sciences community. However, most of the elements presented in this SMP are applicable in other domains. This SMP considers six main software-related sections: documentation; testing; interoperability; community, contribution and governance; reproducibility; and recognition. From these, only the interoperability section has elements unique to the Life Sciences as it focuses on standards agreed within this community. In order to facilitate its understanding, adoption and usage, this SMP has been structured as a series of questions per each section, see Table 1 for more details.

Table 1. Overview of a first draft for research SMPs in Life Sciences.

Section	Questions and options
Documentation	1. What type of documentation is available for the software? (please include URL if available) Options: User documentation, README file, Release notes, Docstring/comments, CHANGELOG, Other, None 2. Is the purpose of the software stated in the documentation? 3. Does the documentation describe how to: test the software, use the software, build the software, deploy the software, install the software
Testing	1. Do you test your software? 2. What type of testing do you use? Options: Unit, Integration, Regression, End-to-end, Other (e.g. linting) 3. Do you use any testing methodology? (e.g. Continuous Integration, Bug-Driven testing, etc.). Please name it. 4. Are the tests for the software automated? 5. Are the tests available with the source code? 6. Do you provide example parameters and input/output data for testing purposes?
Interoperability	1. Do you use existing and standard input/output formats? If yes, please list them and, if possible, include URL.
Community, contributing & governance	1. Does your software have a license? If yes, which one?
Reproducibility	1. Do you use a version control system? If yes, which one? (e.g., Git, Mercurial, Subversion, VCS, Other) 2. Do you assign a version to each release of your software? 3. Do you use Semantic Versioning? 4. How do you define dependencies of your software and their version? For instance Maven (for Java), requirements.txt or environment.yml (Python), package.json (JS) 5. Do you provide input and output examples? Where can they be found?

[4] https://elixir-europe.org/about-us/commissioned-services/software-best-practices

Recognition	1. Do you include citation information (i.e. how to cite your software in the form of citation.cff, codemeta.json or bibtex)? 2. Do the releases have a persistent global unique identifier (such as release on Zenodo with DOI, snapshot or release referenced on Software Heritage with SWH-ID)? 3. Does the citation information contain ORCIDs of the authors

5 Semantic Web and Linked Open Data

The FAIR data principles are directly related to the Semantic Web as they encourage the use of controlled vocabularies following standardized formats such as ontologies. They also extend the Linked Open Data principles [14] adding more specific elements related to, for instance, characteristics of identifiers and preservation [15]. To make the FAIR principles a reality for software, it is necessary to support FAIR software metadata. Some of the controlled vocabularies that can be used to describe software have emerged in recent years. For instance, the Software Ontology [16] helps describe software in the biomedical domain used to store, manage and analyze data. This ontology incorporates terms from the EDAM ontology [17], a vocabulary used to describe data related concepts in the bioinformatics domain, including types of data, formats, topics and operations. With a broader coverage, beyond Life Sciences, the Codemeta project [9] provides metadata elements to describe software including citation elements, versioning, dependencies, purpose, keywords and description. Depending on what domain or community a research software belongs to, one or another vocabulary, or even a mixed of them, will be more appropriate.

Although defining what vocabularies should be used to specify software metadata is out of the scope of the FAIR (software) principles and the SMPs, both of them will have an immediate impact in such vocabularies. For instance, vocabularies might need to include new terms to cover metadata identifying particular software versions (either release or intermediate versions), more specific terms to describe software operations or purposes, or ways to connect software elements, i.e., computational workflows. In addition, SMPs can benefit from FAIR software metadata in a similar way as machine-actionable DMPs do. Machine-actionable DMPs [18] have emerged as a machine-processable version of traditional DMPs, making it easier, for instance, to exchange and compare DMPs across different funding bodies.

6 Final Words

We have presented here two on-going efforts that are cooperating to improve the development and recognition of good practices for research software: FAIR principles and SMPs (particularly for Life Sciences). Although these topics are still under development, they both are being recognized as increasingly important not only to improve research software but also to reinforce the recognition of software as a primary element in research. We invite interested readers to become part of these

efforts and thus collaborate on the development of research practices facilitating more efficient, reproducible and trustworthy results.

References

1. Wilkinson MD, Dumontier M, Aalbersberg IjJ, Appleton G, Axton M, Baak A, et al. The FAIR Guiding Principles for scientific data management and stewardship. Sci Data. 2016;3: 160018. https://doi.org/10.1038/sdata.2016.18
2. Surkis A, Read K. Research data management. J Med Libr Assoc JMLA. 2015;103: 154–156. https://doi.org/10.3163/1536-5050.103.3.011
3. Herzog E, Russell K, Stall S. FAIR Data Maturity Model WG. RDA; 2018 Sep. Available: https://www.rd-alliance.org/groups/fair-data-maturity-model-wg
4. Lamprecht A-L, Garcia L, Kuzak M, Martinez C, Arcila R, Martin Del Pico E, et al. Towards FAIR principles for software. Data Sci. 2019;Preprint: 1–23. https://doi.org/10.3233/DS-190026
5. Hasselbring W, Carr L, Hettrick S, Packer H, Tiropanis T. From FAIR research data toward FAIR and open research software. It - Inf Technol. 2020;62: 39–47. https://doi.org/10.1515/itit-2019-0040
6. Katz DS, Niemeyer KE, Smith AM, Anderson WL, Boettiger C, Hinsen K, et al. Software vs. data in the context of citation. PeerJ Inc.; 2016 Dec. Report No.: e2630v1. https://doi.org/10.7287/peerj.preprints.2630v1
7. Smith AM, Katz DS, Niemeyer KE. Software citation principles. PeerJ Comput Sci. 2016;2: e86. https://doi.org/10.7717/peerj-cs.86
8. Capella-Gutierrez S, Iglesia D de la, Haas J, Lourenco A, Fernández JM, Repchevsky D, et al. Lessons Learned: Recommendations for Establishing Critical Periodic Scientific Benchmarking. bioRxiv. 2017; 181677. https://doi.org/10.1101/181677
9. Jones MB, Boettiger C, Mayes AC, Arfon Smith, Slaughter P, Niemeyer K, et al. CodeMeta: an exchange schema for software metadata. KNB Data Repository. KNB Data Repository; 2016. https://doi.org/10.5063/SCHEMA/CODEMETA-1.0
10. Barker M, Chue Hong N, Psomopoulos F, Garcia Castro LJ, Gruenpeter M, Harrow J, et al. FAIR principles for research software. 2019 Nov. Available: https://www.rd-alliance.org/plenaries/rda-15th-plenary-meeting-australia/fair-principles-research-software
11. Kuzak M, Barker M, Chue Hong N, Garcia Castro LJ, Gruenpeter M, Harrow J, et al. FAIR4RS talk at ECCB2020. 2020 Sep 2. Available: https://docs.google.com/presentation/d/1MvIFKT0Wk0GOLYZ3uF7ZMrO1gb5sFiV5jwIitq76z8A
12. The Software Sustainability Institute. Checklist for a Software Management Plan. Zenodo; 2018 Oct. https://doi.org/10.5281/zenodo.1460504
13. Chue Hong, Neil. Writing and using a software management plan, Software Sustainability Institute. 2014. Available: https://www.software.ac.uk/resources/guides/software-management-plans
14. Berners-Lee, T.. Is your linked open data 5 star. 2009. Available: https://www.w3.org/DesignIssues/LinkedData.html. Retrieved on the 14.Oct.2020
15. Hasnain A., Rebholz-Schuhmann D. (2018) Assessing FAIR Data Principles Against the 5-Star Open Data Principles. In: Gangemi A. et al. (eds) The Semantic Web: ESWC 2018 Satellite Events. ESWC 2018. Lecture Notes in Computer Science, vol 11155. Springer, Cham. https://doi.org/10.1007/978-3-319-98192-5_60
16. Malone J, Brown A, Lister AL, Ison J, Hull D, Parkinson H, et al. The Software Ontology (SWO): a resource for reproducibility in biomedical data analysis, curation and digital

preservation. Journal of Biomedical Semantics. 2014;5: 25. https://doi.org/10.1186/2041-1480-5-25

17. Ison J, Kalas M, Jonassen I, Bolser D, Uludag M, McWilliam H, et al. EDAM: an ontology of bioinformatics operations, types of data and identifiers, topics and formats. Bioinformatics. 2013;29: 1325–1332. https://doi.org/10.1093/bioinformatics/btt113
18. Miksa T, Simms S, Mietchen D, Jones S. Ten principles for machine-actionable data management plans. PLOS Computational Biology. 2019;15: e1006750. https://doi.org/10.1371/journal.pcbi.1006750