FAIR Principles for Research Software (FAIR4RS Principles)

Authors

Neil P. Chue Hong*, Daniel S. Katz*, Michelle Barker*;
Anna-Lena Lamprecht, Carlos Martinez, Fotis E. Psomopoulos, Jen Harrow, Leyla Jael Castro, Morane Gruenpeter, Paula Andrea Martinez, Tom Honeyman;
Alexander Struck, Allen Lee, Axel Loewe, Ben van Werkhoven, Catherine Jones, Daniel Garijo, Esther Plomp, Francoise Genova, Hugh Shanahan, Joanna Leng, Maggie Hellström, Malin Sandström, Manodeep Sinha, Mateusz Kuzak, Patricia Herterich, Qian Zhang, Sharif Islam, Susanna-Assunta Sansone, Tom Pollard, Udayanto Dwi Atmojo;
Alan Williams, Andreas Czerniak, Anna Niehues, Anne Claire Fouilloux, Bala Desinghu, Carole Goble, Céline Richard, Charles Gray, Chris Erdmann, Daniel Nüst, Daniele Tartarini, Elena Ranguelova, Hartwig Anzt, Ilian Todorov, James McNally, Javier Moldon, Jessica Burnett, Julián Garrido-Sánchez, Khalid Belhajjame, Laurents Sesink, Lorraine Hwang, Marcos Roberto Tovani-Palone, Mark D. Wilkinson, Mathieu Servillat, Matthias Liffers, Merc Fox, Nadica Miljković, Nick Lynch, Paula Martinez Lavanchy, Sandra Gesing, Sarah Stevens, Sergio Martinez Cuesta, Silvio Peroni, Stian Soiland-Reyes, Tom Bakker, Tovo Rabemanantsoa, Vanessa Sochat, Yo Yehudi;
and the FAIR4RS WG

(*) Lead authors, with equal contributions

The author list and ordering is determined by the contributions each author has made to the document. The full list of contributors and their contributions can be found in [Appendix C - Contributor List](#).

Abstract

Research software is a fundamental and vital part of research worldwide, yet there remain significant challenges to software productivity, quality, reproducibility, and sustainability. Improving the practice of scholarship is a common goal of the open science, open source software and FAIR (Findable, Accessible, Interoperable and Reusable) communities, but improving the sharing of research software has not yet been a strong focus of the latter.

To improve the FAIRness of research software, the *FAIR for Research Software (FAIR4RS)* Working Group has sought to understand how to apply the *FAIR Guiding Principles for scientific data management and stewardship* to research software, bringing together existing and new community efforts. Many of the FAIR Guiding 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 principles.

This document presents the first version of the *FAIR Principles for Research Software (FAIR4RS Principles)*. It is an outcome of the FAIR for Research Software Working Group (FAIR4RS WG).

The FAIR for Research Software Working Group is jointly convened as an RDA Working Group, FORCE11 Working Group, and Research Software Alliance (ReSA) Task Force.

Date	**Version Number**	**Description**	**Editor(s)**
9/6/2021	0.3	Draft for formal community review	Neil Chue Hong
7/6/2021	0.2.1	Amended abstract and text of F1, F1.1, F1.2, F4 and R1 for review by drafting group	Neil Chue Hong
1/6/2021	0.2	Second draft for review by FAIR4RS Steering Committee	Neil Chue Hong
17/5/2021	0.1	First draft for review by FAIR4RS WG	Neil Chue Hong, Michelle Barker

Table of Contents

Introduction

This document presents the first version of the FAIR Principles for Research Software, for review by the research software community. This work is an outcome of the FAIR for Research Software Working Group (FAIR4RS WG) that convened in 2020 with the aim of developing community-endorsed FAIR principles for research software, which is shortened to the *FAIR4RS Principles* in the remainder of this document.

This document begins with an explanation of the history of this work, starting from the conceptualisation of The FAIR Guiding Principles for scientific data management and stewardship (shortened to the *FAIR Guiding Principles* in the remainder of this document). The proposed FAIR4RS Principles are then provided, alongside an explanation of how they should be interpreted (a comparison of the evolution of these principles is provided in Appendix B). Finally, this document concludes with discussion on the challenges in using and implementing these principles, such as gaps in existing infrastructure and standards that make it hard to follow the principles, and work being undertaken to develop an implementation roadmap to address these.

Extensive community consultation is needed to ensure that the principles developed support the widest possible range of use cases. Feedback has been sought on this document from stakeholders (listed in Appendix C) including people who use research software, write or maintain research software, create or implement policy around research software and other research outputs, manage infrastructure that supports usage or development of research software and/or other research outputs, fund research software and/or other research outputs, and others with an interest in the FAIR4RS Principles. This has resulted in this version of the document, which is now open for formal community review.

Aims

The ultimate goal of FAIR is to enable transparency, reproducibility, and reusability of research. For this to happen, software needs to be well-described (by metadata), inspectable, documented and appropriately structured so that it can be executed, replicated, built-upon, combined, reinterpreted, reimplemented, and/or used in different settings. The FAIR4RS Principles aim to provide guidance to software creators and owners on how to make their software Findable, Accessible, Interoperable and Reusable. The FAIR4RS Principles are also relevant to the larger ecosystem and various stakeholders that support research software (e.g., repositories and registries).

The FAIR4RS WG is jointly convened by the Research Software Alliance (ReSA), Future Of Research Communications and E-Scholarship (FORCE11), and the Research Data Alliance (RDA). The FAIR4RS WG is a global and interdisciplinary community composed of 200+ people who have an interest in the application of FAIR principles to research software and other research outputs, such as software users, software developers and maintainers, policy makers, infrastructure support staff, and funders.

The FAIR4RS WG has worked to define and publish community-endorsed FAIR4RS Principles, to be followed by adoption guidelines and use cases, and this document is part of this process. This effort requires extensive consultation with the research community to create the principles and encourage their adoption. The resulting adoption and implementation of FAIR4RS principles will create significant outcomes for many stakeholders, ranging from increased research reproducibility for research organizations, to clarity for funders around their own requirements for software investments, and guidelines for publishers on sharing requirements.

Previous work

It is useful to understand the origins of the concept of FAIR when considering its application to software, and the FAIR4RS WG builds on previous efforts for both FAIR in general, and FAIR research software.

The concept of FAIR originated in the Netherlands during the 2014 Lorentz Workshop "Jointly Designing a Data FAIRport", where participants formulated the FAIR data vision to optimize data sharing and reuse by humans and machines. This vision supports existing communities that try to realize and enable a situation where valuable scientific data is 'FAIR' in the sense of being Findable, Accessible, Interoperable and Reusable. The resulting Guiding Principles can be viewed and commented on (FORCE11, 2015) and resulted in the publication of "The FAIR Guiding Principles for scientific data management and stewardship" (Wilkinson et al., 2016).

From the outset, the FAIR Guiding Principles were intended to be applicable to many kinds of digital assets, not just datasets. A number of research communities and groups have been considering how to apply aspects of FAIR to research software since 2017. Community-produced outcomes before February 2020 can be found in the Software Source Code identification Interest Group's Wiki FAIR4Software reading resources. Newer resources can be found in the FAIR4RS collection on Zenodo and the literature review completed by the FAIR4RS subgroup (FAIR4RS WG, 2021). General work on FAIR has also recognized the need to incorporate other digital objects, and some recent works have specifically highlighted the need for inclusion of software (e.g., European Commission, 2018, European Commission & EOSC Executive Board, 2020, Gruenpeter et al., 2020).

The FAIR4RS WG coordinated four subgroups from July 2020 to March 2021 to provide outputs to support the development of the FAIR4RS Principles:

- A fresh look at FAIR for Research Software examined the FAIR Guiding Principles in the context of research software from scratch, not based on pre-existing work (Katz, Gruenpeter & Honeyman, 2021).
- FAIR work in other contexts examined efforts to apply FAIR Guiding Principles to different forms including workflows, notebooks and training material, to provide insights for the definition and implementation of FAIR principles for research software.
- Defining Research Software: a controversial discussion reviews existing definitions of research software in order to provide the overall context of the subgroup outputs (Gruenpeter et al., 2021).
- Review of new research related to FAIR Software that has come out since the release of "Towards FAIR principles for research software" (Lamprecht et al., 2020) and review of the principles set out in that paper (Chue Hong et al., 2021).

The work of the subgroups was brought together and presented for consultation by the wider FAIR4RS community (Katz, Chue Hong, Barker & Gruenpeter, 2021).

Development of the FAIR4RS Principles

This section explains the context within which the FAIR4RS Principles should be understood, based on the input received by the FAIR4RS WG and subgroups, as part of the first community consultation, and the subsequent discussion and development. In proposing these FAIR principles for research software, the intent and methods of the FAIR Guiding Principles were taken as the starting point: to "maximize the added-value gained by contemporary, formal scholarly digital publishing" and "to ensure transparency, reproducibility, and reusability." Importantly, it is acknowledged that the foundational principles of *Findable, Accessible, Interoperable, and Reusable* may need to be reinterpreted to both stay true to the goals of the FAIR Guiding Principles but also ensure that they are applicable to software. It is also recognized that the goal of the FAIR Guiding Principles, and the FAIR4RS Principles is to support both human-driven and machine-driven activities.

The FAIR principles are aspirational, and the use of indicators should be to show progression to increasing FAIRness. FAIR is not binary, nor should the number of principles that something implements be used as a direct numerical comparison of how FAIR something is compared to another object.

In a perfectly FAIR world, third-party code that FAIR software has dependencies on would ideally be FAIR as well. But, because software consists of large stacks of interdependent components, any definition of metrics and indicators of FAIR for software can only be made in the context of specific components with which it is designed to work, not all software with which it could be combined.

The definition of software can include source code, executables or other forms that make sense. Different forms may benefit different users: for a research software engineer, source code may best enable reusability; whereas, for a researcher, the executable may be more reusable. However, the source code is often the most useful form to understand the software, and the easiest form to apply the FAIR4RS Principles.

FAIR should be applied to things which are published or shared in some way. These are variously referred to in other literature as research objects, research outputs, research artefacts, research assets, digital research objects, digital assets, digital research artefacts, non-data assets, scholarly outputs, digital objects, etc. The FAIR4RS Principles should therefore be applied to any and all research software that underpins published findings, including software used as a tool, software produced as a research outcome or results, or software which is itself the object of research.

The application of the FAIR4RS Principles to software does not depend on the long-term preservation of software. However, software has a wide range of useful lifetimes, and the ability of the software to serve as a record of work done by researchers will degrade over time if the software is not adequately preserved.

The FAIR4RS Principles can be applied to any software intended to be used in research, but there is no value in trying to define the “boundary” of what counts as software used in research (e.g., closed vs. open, commercial vs. not-for-profit, research software vs. academic software vs. non-academic software) in terms of interpreting the FAIR Guiding Principles for software.

Many software engineering practices are relevant to various of the FAIR4RS Principles. For instance: localization can improve accessibility, design patterns can improve interoperability, and documentation and encapsulation can improve reusability. Nevertheless, while important more generally for producing high quality software, they are best addressed separately from (but as a complement to) the FAIR4RS Principles.

Finally, and importantly, the application of the FAIR4RS Principles is the responsibility of the owners (who are often the creators) of the software, not the users. Responsibility can be shared in the long-term with other stewards and also depends on an ecosystem that enables this, e.g., appropriate repositories and registries. This must be emphasized, as those producing the software are best placed to ensure they provide the necessary information to make their work as FAIR as possible, and get credit for doing so in return.

FAIR Principles for Research Software

In this section, each of the FAIR4RS Principles is proposed and explained. First, each foundational principle (F, A, I and R) is described, followed by the numbered guiding principles used to interpret the foundational principle.

- Text in **bold** is the text for the principles.
- Text in *italics* is the narrative text explaining the intent behind the phrasing of the principles and providing guidance for how they should be interpreted.

A key challenge of defining these FAIR principles for research software is the balance between aspirational principles vs actionable principles. In line with the FAIR Guiding Principles, the FAIR4RS are intended to be aspirational. For this reason indicators are also needed, not just to document the level of FAIRness of the software at a given moment in time, but to provide a clear and measurable path to progressively improve the FAIRness of the software. This will benefit the transparency, reproducibility and reusability of research. How a principle can be implemented in an actionable way will be described in the guidance that will be produced as the next stage of the FAIR4RS WG work.

Findable

F: The software, and its associated metadata, should be easy to find for both humans and machines.

Machine-readable metadata are essential for automatic discovery of software and this metadata should meet domain-relevant community standards.

F1. Software is assigned a globally unique and persistent identifier.

The use of globally unique and persistent identifiers enable adherence to many of the other FAIR4RS Principles by removing ambiguity (for humans and machines) around what software (or part of it) is being referenced. Complexities around granularity (the “level of detail being implemented”) and software versions (the “changes between implementations”) are addressed by F1.1 and F1.2.

F1.1. Different components of the software must be assigned distinct identifiers representing different levels of granularity.

When assigning identifiers to different granularity levels or parts of the same software, each component must have a different identifier. The relationship between components should be embodied in the associated metadata. This allows for an identifier at a higher conceptual level for the software to be related to its subcomponents.

Granularity levels for software are shown in Figure 1 in Appendix A. These principles do not prescribe which granularity levels should be assigned identifiers, as this is likely to be implementation-specific. Nevertheless, it is important to acknowledge the relationship between the different granularity levels and the types of identifier most suited for each case.

F1.2. Different versions of the same software must be assigned distinct identifiers.

To make different versions of the same software (or component) findable, each version must have a different identifier. The relationship between versions of components should be embodied in the associated metadata. What is considered a "version" is defined by the owner of the software: in most cases this will be something which is "released" or "published" so that others can use it.

There are existing software engineering practices (e.g. version control, semantic versioning) around the management and versioning of software that may form part of the implementation of these relationships .

It is important to understand how one version of a piece of software relates to another, particularly as metadata such as authors or name/title may change. Although most identifier systems support relations that can be used to implement this functionality, some repositories do not yet, and this should not discourage the application of identifiers to software because of a requirement to support versioning.

F2. Software is described with rich metadata.

Metadata should be used to support search and discoverability (including indexing). This principle means that software has descriptive metadata. This metadata must itself be FAIR (F4), and should follow community standards and use controlled vocabularies. The FAIR4RS principles do not define which standards should be used, as this is better captured in guidance for implementing the principles coming out of each community. Principles R1, R1.1 and R1.2 describe categories of metadata that enable reuse.

F3. Metadata clearly and explicitly include the identifier of the software they describe.

The association between the metadata (wherever it is stored, F4) and the software should be made explicit by mentioning the software's globally unique and persistent identifier in the metadata. In conjunction with A1, this means the metadata describe how the software can be obtained. For software to be findable, metadata are not required to include references for all of its dependencies. Principles I2 and R2 describe how references to dependencies make software interoperable and reusable.

F4. Metadata are FAIR and are searchable and indexable.

Making the metadata about the software FAIR, making it discoverable by both humans and machines, supports the findability of software by supporting searching and indexing by others. It allows the metadata to be published in or harvested by a registry or catalog, or by a search engine. Publishing software metadata in FAIR repositories improves the discovery of software by making it available in a searchable resource.

Accessible

A: The software, and its metadata, must be retrievable via standardized protocols.

In the FAIR Guiding Principles, accessibility translates into retrievability. In these FAIR4RS Principles, accessibility is also narrowly scoped to the ability to "retrieve" the software. Because software by necessity requires the use of standardized communications protocols to operate, some of the FAIR Guiding Principles may be considered commonly understood and implemented for software.

For software to be accessible, it may be made available in any form (including, but not limited to source code, executable, library, or service) as long as the conditions for access are clearly stated and transparent. The software should be retrievable by both humans and machines.

For software, "accessibility" can also be used to refer to the ability to access or use the software regardless of impairment. While this is important for research software, these are best addressed separately from (but as a complement to) the FAIR4RS Principles.

A1. Software is retrievable by its identifier using a standardized communications protocol.

Different types of software have different methods for access. For instance, software that is only available in source code form may be downloaded from a repository before being compiled locally, whereas software hosted as a service on a remote server may be accessed without retrieving it. This principle states that to obtain the software should not require specialised or proprietary tools or communication methods.

A1.1. The protocol is open, free, and universally implementable.

It is the openness of the protocol (including the resolver for the identifier) that is important, not the implementation of the infrastructure that supports it. Here "open" means that there are no restrictions to implementing it and "free" means that there are no fees or licensing costs to implement it.

A1.2. The protocol allows for an authentication and authorization procedure, where necessary.

There are often conditions of access to software, for instance agreeing to a license, requirement for payment or restrictions based on the privilege level of the user. This procedure may be a manual one, for instance requiring the signing of a non-disclosure agreement, researcher security check, or software having an embargo period, however this is not optimal as the FAIR principles put specific emphasis on enhancing the ability of machines to use digital objects. It may also include things such as requiring a license server to be contacted.

A2. Metadata are accessible, even when the software is no longer available.

Availability of software may change over time, because there is a cost to maintaining access or because the software has degraded and is no longer safely usable. The metadata describing the software is generally easier and cheaper to store and maintain than the software itself (e.g. in the software repository, or in a software registry or catalog) and there is value in understanding the details of the software even if it is no longer accessible.

Interoperable

I: The software interoperates with other software through exchanging data and/or metadata, and/or through interaction via application programming interfaces (APIs).

The definitions of interoperability and reusability as defined by the FAIR Guiding Principles overlap when applied to software. To differentiate between the two, interoperability is limited to being concerned with the capacity to exchange data between independent (i.e. can be executed separately) software. As an example, the sense of “integrated” that applies to data (where two pieces of data combine to form a new third thing) does not apply in the same way to software where, in a sense, all software is “integrated” with, or depends on, other software (and this concept is more sensibly placed under reusability, in R2). Software also has “agency”: software calls on other software. Two independent pieces of software can be said to interoperate when the functionality exists in both to read and write or otherwise exchange the same formats.

I1. Software reads, writes and exchanges data in a way that meets domain-relevant community standards.

Software interoperates through the exchange of data. This includes the use of data types and formats that ideally are formally described using controlled vocabularies, to facilitate machine readability and data exchange. Examples of community standards for data are curated by the FAIRSharing Registry at https://fairsharing.org/standards/.

Where software interacts via APIs, these should be documented so that their capabilities can be inspected and understood by humans and machines.

I2. Software includes qualified references to other objects.

Some software includes references to external data objects required to execute the software (e.g. parameter files for certain applications). To be fully FAIR, the data would ideally be FAIR as well, and references to external data fully qualified. Qualified references should be to metadata and data, as well as to non-digital objects that have a virtual presence in digital systems (e.g., samples, reagents, etc.), which with the software interacts. These qualified references should be described using identifiers and/or controlled vocabularies. "Qualified" means specifying the authoritative source for an identifier or vocabulary item, possibly including a resolvable reference to further information about the source.

Reusable

R: The software is both usable (it can be executed) and reusable (it can be understood, modified, built upon, or incorporated into other software).

The definitions of interoperability and reusability as defined by the FAIR Guiding Principles overlap when applied to software. To differentiate between the two, reusability (implicitly including usability) focuses on the ability of humans and machines to execute, inspect and understand the software, so it can be modified, built upon, or incorporated into other software.

Note that the general intent of these principles is that software is "executable in principle" - not "guaranteed to execute". Also, different aspects of reusability may best apply to different forms of software. For instance, source code might be modifiable but not executable without specialist infrastructure; libraries available as binaries can be built on and incorporated into other software but not easily modified. In general, source code is the most reusable form of software.

R1. Software is described with a plurality of accurate and relevant attributes.

It is easier to reuse (and find) software if there are many descriptive labels attached to it. F2 requires software to be described by rich metadata. Software should be described for the categories of R1.1 (license), R1.2 (provenance), and additionally address the categories of metadata that facilitate reuse. Relevant attributes can be determined by repositories, and by communities who create and reuse software. Plurality means that, where possible, multiple terms for the same, similar or overlapping concepts should be provided to enable the broadest possible reuse.

R1.1. Software must have a clear and accessible license.

Software must have a license that clearly describes how it can be used and reused, ideally with conditions that are clear to humans and machines (e.g. using the specification published by the SPDX Consortium, 2020). To support a wide range of reuse scenarios, the license should be as open as possible. This license must also be compatible with the requirements of the licenses of the software's dependencies so that the software can be legally combined.

R1.2. Software is associated with detailed provenance.

The primary provenance information for software is authorship. Although a version control system may provide detailed provenance for software, care must be taken that authorship and contributions are properly tracked, as other types of contributions may not be recorded by the commit history. Provenance is also important because it provides verifiable information regarding the origin and development history of software, such as where it was developed, what standards were applied, and which tools and settings were used. This can be used to understand design decisions, or constraints and limitations to reusability.

R2. Software includes qualified references to other software.

Software is rarely standalone and in most cases is built upon other software (e.g dependencies), it should include appropriate references to other software (requirements, imports, libraries, etc.) which is necessary to compile and run the software. "Qualified" here means specifying the authoritative source for an identifier, possibly including a resolvable reference to further information about the source. To follow this principle, it is desirable but not required that the other software referenced implements the FAIR4RS Principles.

R3. Software meets domain-relevant community standards.

Software, including its documentation and license, should meet or rise above domain-relevant community standards and coding practices (for example choice of programming language, standards for testing, usage of file formats, etc) that enable reuse. While the FAIR4RS Principles do not specify particular community standards, the intent is to ensure that practitioners are aware of what others are doing and using in the community, e.g. through initiatives like FAIRsharing (Sansone et al, 2019), whilst acknowledging that community standards are (and should be) under constant development.

This principle is not intended to stifle innovation in the development of software, but discourage fragmentation of community practice. If a piece of software aims to rise above existing community standard or coding practice, for instance by using a different programming language or file format that better enables research in the domain, an eventual outcome should be a revised or new standard or practice being adopted by the community.

Challenges to implementation

In this section, challenges that must be overcome to make research software FAIR are described. These represent current gaps that may make it difficult for individuals to follow the FAIR4RS Principles. Additional details can be found in the related reports (Katz, Gruenpeter & Honeyman, 2021; Katz, Chue Hong, Barker & Gruenpeter, 2021).

Metadata and identifier authority. All research software must have unique identifiers and associated metadata. How are these identifiers created? How is the metadata created, stored and maintained? Intrinsic metadata, such as a codemeta.json file in the source code repository, is guaranteed to be controlled by the authors but must be exposed to make the software findable. Extrinsic metadata, such as (persistent) identifiers, can be used to make the software findable but is controlled by an external authority.

Metadata vocabularies and metadata properties. At present, there is no community agreement on which vocabularies should be used. Vocabularies used by package managers to describe software do not capture metadata about research and there are relatively few discipline-specific vocabularies that capture metadata about software development and usage. Establishing metadata vocabularies/standards is an intensive process for which resources are limited.

Software identifiers. At present, there is no community agreement on the best identifiers for software, even for specific use cases such as giving software authors credit. These identifiers are mostly independent and not clearly interoperable. This could be partly addressed through a community endorsement process, in one or more relevant communities.

Identification target. At present, there is no community agreement on what a software identifier should refer to, e.g. for open source software, for commercial software, for a container, for a service, etc. This is discussed in the FAIR4RS Principles when talking about granularity and versions, and is also related to the idea of a software concept, which is the set of all specific versions of that software. Other work, such as (Hata et al., 2021), has highlighted challenges related to the linkage of scientific knowledge and software artifacts, e.g. in executable.

Software structure complexity. Software is often a complex object made up of other software, documentation, data and metadata, whose versions may change at different rates. How is this dealt with? Where should the FAIR4RS Principles be applied, and where should other interpretations of the FAIR Guiding Principles be applied? What should have identifiers, and how should relationships between them be described to be FAIR? Here, experience from applying the FAIR Guiding Principles to complex data collections may provide solutions.

FAIRness of related research objects. There is still debate over whether FAIR is recursive, i.e. a digital research object is only “fully FAIR” if the objects it builds on are also FAIR. However, even if just applied to data dependencies, this would restrict the implementation of FAIR4RS Principles as it would require measurable, actionable guiding principles to be applied down the complete dependency stack. This would ultimately be intractable as the authors of the software would not have responsibility for making the depende

Definition of accessibility. In software engineering, there is already a different, well-understood definition of software accessibility. Even if the meaning used in the FAIR4RS Principles is well-defined and scoped, it may lead to confusion and mean the principle is not well-understood across all domains.

Definition of reusability. In software engineering, for software to be reusable it should also be maintainable and dependable (able to be built on for other purposes). This may be captured in R3, around domain-relevant community standards, but may also require additional clarification to avoid confusion or the proliferation of many competing sets of “added letters” to FAIR4RS related to other qualities.

Openness and FAIR. Software does not need to be open source licensed to be FAIR: FAIR ≠ Open, Nevertheless, for software it may be easier to make it FAIR if it is open source. This can be seen in Figure 2 in Appendix A which summarizes software as increasingly FAIR research objects. Open source software is generally more reusable, as the source code is accessible, and may be more interoperable because its APIs are inspectable. As with data, FAIR software should strive to be “as open as possible, as closed as necessary”.

The path to adoption

It will take significant effort to gain wide-spread adoption of the FAIR4RS Principles, once finalized. The convening of the FAIR4RS WG across RDA, FORCE11, and ReSA will support usage of the outcomes across those communities. RDA, FORCE11, and ReSA will systematically promote the outcomes, aiming to raise awareness and facilitate a wider adoption of the FAIR4RS WG outcomes by existing and emerging initiatives. Organizations with a focus on FAIR will also be engaged, to encourage promotion of the application of FAIR to research outputs other than data. For example, the FAIRsFAIR project has produced a set of recommendations for the creation of FAIR guiding principles for research software (Gruenpeter et al., 2020) and the FAIR4RS Principles will be assessed against these.

The FAIR4RS WG's aims also include development of adoption guidelines and practices to enable widespread adoption of the FAIR4RS Principles across the research software community at national, disciplinary, and international levels. This will focus on the needs of a variety of stakeholders, including:

- those that will endorse and promote the guidelines
- those that will provide training on the guidelines
- users of the guidelines

The FAIR4RS WG will focus on adoption after the dissemination of the FAIR4RS principles in mid-2021, and will continue to regularly engage the community during all phases.

ReSA is also leading the FAIR4RS Roadmap to make FAIR research software a reality, with support from the Wellcome Trust. The 2018 European Commission report, "Turning FAIR into Reality" (European Commission, 2018), concludes that FAIR digital objects (including software) need to be supported by metrics, incentives, skills and FAIR services that provide persistent identifiers, metadata specifications, stewardship and repositories, actionable policies and Output Management Plans. All of these need to be created for FAIR software, to complement the significant FAIR initiatives that primarily encompass data, and to leverage the efforts already underway to enable this for software. The FAIR4RS Roadmap is identifying relevant software initiatives and equivalent FAIR data programs in areas such as: indicators, metrics, maturity models and certification; curriculums and competence centres, career profiles and reward structures; certification of FAIR services; interoperability frameworks; and policy change.

Acknowledgements

This work was partially supported by the Alfred P. Sloan Foundation grant G-2021-14116 and Wellcome Trust grant 222436/Z/21/Z. NCH's time as editor was supported by the UKRI research councils through grant EP/S021779/1.

References

Chue Hong, N., Lamprecht, A-L., Desinghu, B., Busse, C., Garijo, D., Plomp, E., Giacomoni, F., Harrow, J., Molden, J., Johnston, K., Garcia Castro, L.J., Hellstrom, M., Barker, M., Meyers, N., Martinez, P.A., Herterich, P., Zhang, Q., Gesing, S., Martinez Cuesta, S., Honeyman, T. (2021). What makes software FAIR? Reviewing the Towards FAIR Principles for Research Software paper and new research related to FAIR software: A report from FAIR4RS Subgroup 4. Zenodo. https://doi.org/10.5281/zenodo.4908919

European Commission. Directorate General for Research and Innovation. (2018). Turning FAIR into reality: final report and action plan from the European Commission expert group on FAIR data. Publications Office. https://doi.org/10.2777/1524

European Commission. Directorate General for Research and Innovation. & EOSC Executive Board,. (2020). Six Recommendations for implementation of FAIR practice by the FAIR in practice task force of the European open science cloud FAIR working group. Publications Office. https://doi.org/10.2777/986252

FAIR4RS WG. (2021). FAIR4RS Subgroup 4 - reading list of new research (Version 1.0) [Data set]. Zenodo. http://doi.org/10.5281/zenodo.4555865

FORCE11 FAIR Data Publishing Working Group. (2015). The FAIR Data Principles. FORCE11. https://www.force11.org/group/fairgroup/fairprinciples

GO FAIR. (2018). FAIR Principles. GO FAIR Initiative. https://www.go-fair.org/fair-principles/ Retrieved from:
https://web.archive.org/web/20180212143802/https://www.go-fair.org/fair-principles/

Gruenpeter, M., Di Cosmo, R., Koers, H., Herterich, P., Hooft, R., Parland-von Essen, J., Tana, J., Aalto, T., & Jones, S. (2020). M2.15 Assessment report on 'FAIRness of software' (Version 1.1). Zenodo. https://doi.org/10.5281/zenodo.4095092

Gruenpeter, M., Katz, D.S., Lamprecht, A-L., Honeyman, T., Garijo, D., Struck, A., Niehues, A., Martinez, P.A., Castro, L.J., Rabemanantsoa, T., Plomp, E., Chue Hong, N., Martinez-Ortiz, C., Sesink, L., Liffers, M., Fouilloux, A.C., Erdmann, C., Peroni, S., Lavanchy, P.M., & Todorov, I. (2021). Defining Research Software: a controversial discussion: Summary Report of FAIR4RS Subgroup 3 activity and discussion. Manuscript in preparation.

Hata, H., Guo, J. L., Kula, R. G., & Treude, C. (2021). Science-Software Linkage: The Challenges of Traceability between Scientific Knowledge and Software Artifacts. *arXiv preprint arXiv:2104.05891*.

Katz, D. S., Gruenpeter, M., & Honeyman, T. (2021). Taking a fresh look at FAIR for research software. Patterns, 2(3), 100222. https://doi.org/10.1016/j.patter.2021.100222

Katz, Daniel S., Chue Hong, Neil P., Barker, Michelle, & Gruenpeter, Morane. (2021). FAIR4RS WG subgroup community consultation March 2021. Zenodo. http://doi.org/10.5281/zenodo.4635410

Lamprecht, A.-L., Garcia, L., Kuzak, M., Martinez, C., Arcila, R., Martin Del Pico, E., Dominguez Del Angel, V., van de Sandt, S., Ison, J., Martinez, P. A., McQuilton, P., Valencia, A., Harrow, J., Psomopoulos, F., Gelpi, J. Ll., Chue Hong, N., Goble, C., & Capella-Gutierrez, S. (2020). Towards FAIR principles for research software [JB]. Data Science, 3(1), 37–59. https://doi.org/10.3233/DS-190026

Research Data Alliance/FORCE11 Software Source Code Identification WG, Allen, A., Bandrowski, A., Chan, P., Di Cosmo, R., Fenner, M., Garcia, L., Gruenpeter, M., Jones, C. M., Katz, D. S., Kunze, J., Schubotz, M. & Todorov, I. T. (2020). Use cases and identifier schemes for persistent software source code identification (V1.1). Research Data Alliance. https://doi.org/10.15497/RDA00053

Sansone, S.-A., McQuilton, P., Rocca-Serra, P., Gonzalez-Beltran, A., Izzo, M., Lister, A.L. and Thurston, M. (2019) FAIRsharing as a community approach to standards, repositories and policies. Nature biotechnology, 37, 358: https://doi.org/10.1038/s41587-019-0080-8

SPDX Consortium. (2020). The Software Package Data Exchange (SPDX®) Specification Version 2.2. Linux Foundation. Retrieved from: https://spdx.github.io/spdx-spec/

Wilkinson, M. D., Dumontier, M., Aalbersberg, Ij. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., Gonzalez-Beltran, A., Gray, A.J.G., Groth, P., Goble, C., Grethe, J.S., Heringa, J., ’t Hoen, P.A.C., Hooft, R., Kuhn, T., Kok, R., Kok, J., Lusher, S.J., Martone, M.E., Mons, A., Packer, A.L., Persson, B., Rocca-Serra, P., Roos, M., van Schaik, R., Sansone, S-A., Schultes, E., Sengstag, T., Slater, T., Strawn, G., Swertz, M.A., Thompson, M., van der Lei, J., van Mulligen, E., Velterop, J., Waagmeester, A., Wittenburg, P., Wolstencroft, K., Zhao, J. & Mons, B. (2016). The FAIR Guiding Principles for scientific data management and stewardship. Scientific Data, 3(1). https://doi.org/10.1038/sdata.2016.18

Appendices

Appendix A - Additional Figures

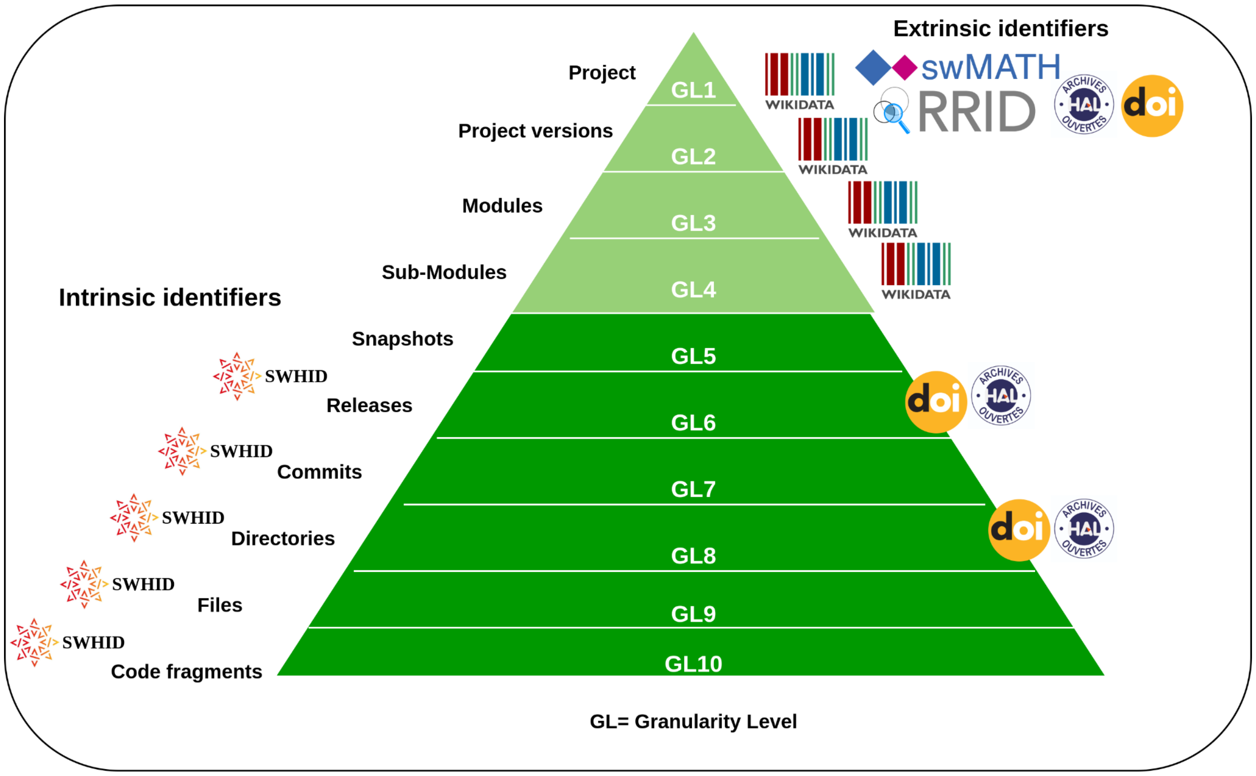

Figure 1: Granularity levels and identifiers currently in use for software [based on granularity levels definition from (RDA/FORCE11 SSCID WG et al., 2020)]

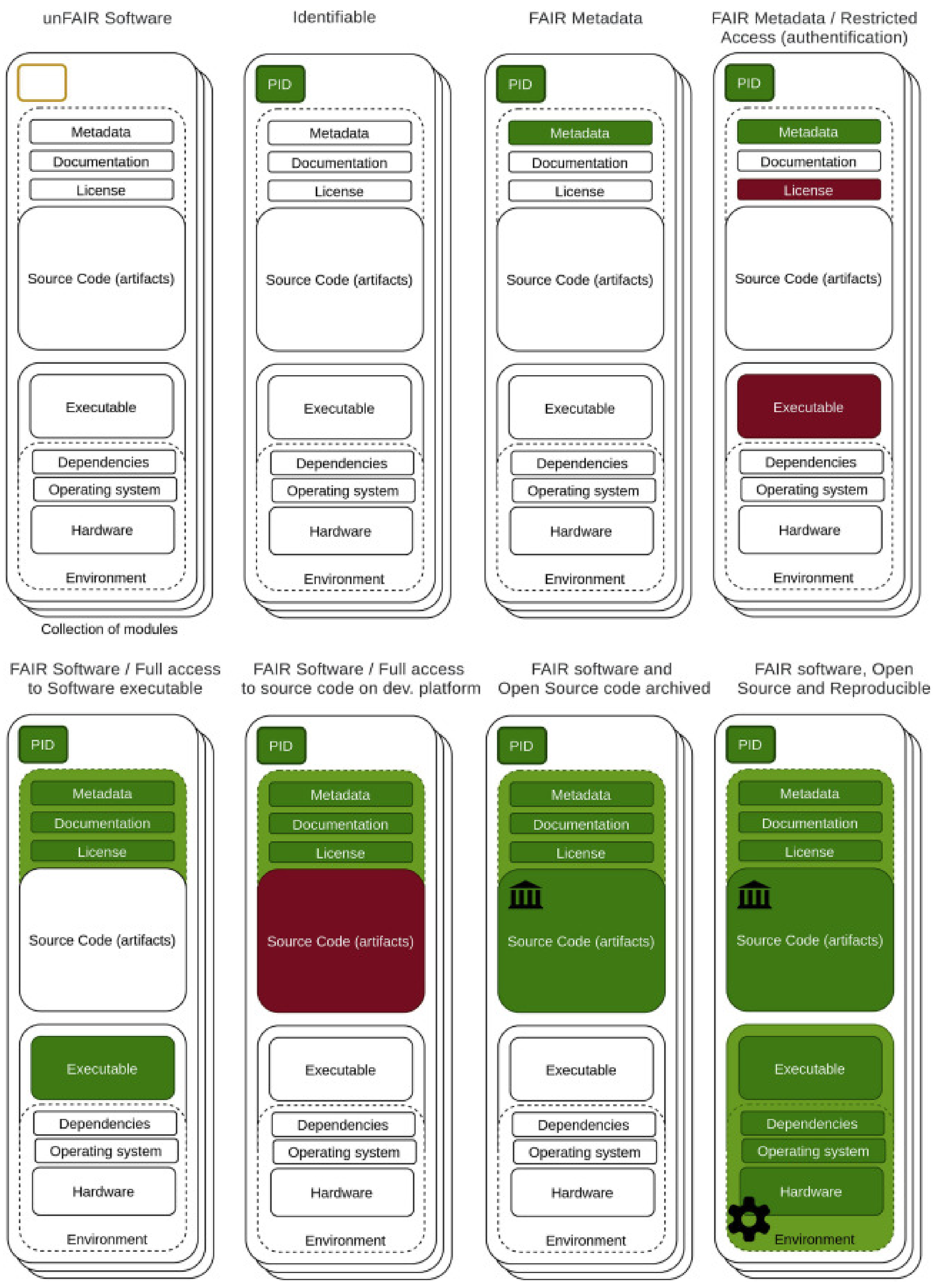
unFAIR Software
Identifiable
FAIR Metadata
FAIR Metadata / Restricted Access (authentification)
PID
Metadata
Documentation
License
Source Code (artifacts)
Executable
Dependencies
Operating system
Hardware
Environment
Collection of modules
FAIR Software / Full access to Software executable
FAIR Software / Full access to source code on dev. platform
FAIR software and Open Source code archived
FAIR software, Open Source and Reproducible

Figure 2: Summarizing software as increasingly FAIR research objects (Katz, Gruenpeter & Honeyman, 2021)

Appendix B - Comparison of FAIR Principles

As background information, this section details how the development of the FAIR4RS Principles has evolved, by comparison of The FAIR Guiding Principles for scientific data management and stewardship (Wilkinson et al., 2016, with foundational principle text taken from GO FAIR, 2018) with the Towards FAIR Principles for research software (Lamprecht et al., 2020) and Taking a fresh look at FAIR for research software report (Katz, Gruenpeter & Honeyman, 2021), and the FAIR4RS Principles described in this document.

FAIR Guiding Principles (2016)	**Towards FAIR Principles for research software (2020)**	**Taking a fresh look at FAIR for research software (2021)**	**FAIR4RS Principles (2021)**
F. Findable			
The first step in (re)using data is to find them. Metadata and data should be easy to find for both humans and computers. Machine-readable metadata are essential for automatic discovery of datasets and services, so this is an essential component of the FAIRification process.	The main concern of findability for research software is to ensure software can be identified unambiguously when looking for it using common search strategies.	The first step in (re)using software is to find it. Metadata and software should be easy to find for both humans and computers. Machine-readable metadata are essential for automatic discovery of software, so this is an essential component of the FAIRification process.	The software, and its associated metadata, should be easy to find for both humans and machines.
F1. (Meta)data are assigned a globally unique and persistent identifier	F1. Software and its associated metadata have a global, unique and persistent identifier for each released version.	F1. Software is assigned a globally unique and persistent identifier	F1. Software is assigned a globally unique and persistent identifier.
			F1.1. Different components of the software must be assigned distinct identifiers representing different levels of granularity.
			F1.2. Different versions of the same software must be assigned distinct identifiers.

F2. Data are described with rich metadata (defined by R1 below)	F2. Software is described with rich metadata.	F2. Software is described with rich metadata (defined first by R1 below, and then by the original FAIR principles for metadata)	F2. Software is described with rich metadata.
F3. Metadata clearly and explicitly include the identifier of the data they describe	F3. Metadata clearly and explicitly include identifiers for all the versions of the software it describes.	F3. Metadata clearly and explicitly include the identifier of the software they describe	F3. Metadata clearly and explicitly include the identifier of the software they describe.
F4. (Meta)data are registered or indexed in a searchable resource	F4. Software and its associated metadata are included in a searchable software registry.	F4. Software is registered or indexed in a searchable resource	F4. Metadata are FAIR and is searchable and indexable.
A. Accessible			
Once the user finds the required data, she/he needs to know how can they be accessed, possibly including authentication and authorisation.	Accessibility translates into retrievability [...] however, we found mere retrievability not enough. In order for anyone to use any research software, a working version of the software needs to be available.	Once the user finds the required software, they need to know how it can be accessed, possibly including authentication and authorization.	The software, and its metadata, must be retrievable via standardized protocols.
A1. (Meta)data are retrievable by their identifier using a standardized communications protocol	A1. Software and its associated metadata are accessible by their identifier using a standardized communications protocol.	A1. Software is retrievable by its identifier using a standardized communications protocol	A1. Software is retrievable by its identifier using a standardized communications protocol.
A1.1. The protocol is open, free, and universally implementable	A1.1. The protocol is open, free, and universally implementable.	A1.1. The protocol is open, free, and universally implementable	A1.1. The protocol is open, free, and universally implementable.
A1.2. The protocol allows for an authentication and authorization procedure, where necessary	A1.2. The protocol allows for an authentication and authorization procedure, where necessary.	A1.2. The protocol allows for an authentication and authorization procedure, where necessary	A1.2. The protocol allows for an authentication and authorization procedure, where necessary.
A2. Metadata are accessible, even when the data are no longer	A2. Software metadata are accessible, even when the	A2. Metadata are accessible, even when the software is no	A2. Metadata are accessible, even when the software is no

available	software is no longer available.	longer available	longer available.
I. Interoperable			
The data usually needs to be integrated with other data. In addition, the data need to interoperate with applications or workflows for analysis, storage, and processing.	Interoperability for research software can be understood in two dimensions: as part of workflows (horizontal dimension) and as stack of digital objects that need to work together at compilation and execution times (vertical dimension)	The software usually needs to communicate with other software via exchanged data (or possibly its metadata). Software tools can interoperate via common support for the data they exchange.	The software interoperates with other software through exchanging data and/or metadata, and/or through interaction via application programming interfaces (APIs).
I1. (Meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation.	I1. Software and its associated metadata use a formal, accessible, shared and broadly applicable language to facilitate machine readability and data exchange.	I1. Software should read, write or exchange data in a way that meets domain-relevant community standards	I1. Software reads, writes and exchanges data in a way that meets domain-relevant community standards.
I2. (Meta)data use vocabularies that follow FAIR principles	I2.1. Software and its associated metadata are formally described using controlled vocabularies that follow the FAIR principles.		
	I2.2. Software use and produce data in types and formats that are formally described using controlled vocabularies that follow the FAIR principles.		
I3. (Meta)data include qualified references to other (meta)data		I2. Software includes qualified references to other objects.	I2. Software includes qualified references to other objects.
	I4S. Software dependencies are documented and mechanisms to access them exist.		

R. Reusable			
The ultimate goal of FAIR is to optimize the reuse of data. To achieve this, metadata and data should be well-described so that they can be replicated and/or combined in different settings.	Reusability in the context of software has many dimensions. At its core, reusability aims for someone to be able to reuse software reproducibly.	The ultimate goal of FAIR is to enable and encourage the use and reuse of software. To achieve this, software should be well-described (by metadata) and appropriately structured so that it can be replicated, combined, reinterpreted, reimplemented, and/or used in different settings.	The software is both usable (it can be executed) and reusable (it can be understood, modified, built upon, or incorporated into other software).
R1. (Meta)data are richly described with a plurality of accurate and relevant attributes	R1. Software and its associated metadata are richly described with a plurality of accurate and relevant attributes.	R1. Software is richly described with a plurality of accurate and relevant attributes	R1. Software is described with a plurality of accurate and relevant attributes.
R1.1. (Meta)data are released with a clear and accessible data usage license	R1.1. Software and its associated metadata have independent, clear and accessible usage licenses compatible with the software dependencies.	R1.1. Software is made available with a clear and accessible software usage license	R1.1. Software must have a clear and accessible license.
R1.2. (Meta)data are associated with detailed provenance	R1.2. Software metadata include detailed provenance, detail level should be community agreed.	R1.2. Software is associated with detailed provenance	R1.2. Software is associated with detailed provenance.
R1.3. (Meta)data meet domain-relevant community standards	R1.3. Software metadata and documentation meet domain-relevant community standards.	R1.3. Software meets domain-relevant community standards	R3. Software meets domain-relevant community standards.
		R2. Software includes qualified references to other software	R2. Software includes qualified references to other software.

Appendix C - Contributor List

The following table lists all people who have been recorded as having made a significant contribution towards the development of the FAIR4RS Principles, listed in alphabetical order by first name. If your contribution has not been properly recognized, please contact N.ChueHong@software.ac.uk.

Name	Institution	ORCID	Editor	Member of drafting group	Member of WG steering committee	Contributor to WG meetings	Contributor to subgroup reports	Contributor to first consultation	Contributor to second consultation
Alan Williams	The University of Manchester	0000-0003-3156-2105					X		
Alexander Struck	Cluster of Excellence Matters of Activity at Humboldt-Universitaet zu Berlin	0000-0002-1173-9228				X	X		X
Allen Lee	Arizona State University	0000-0002-6523-6079							X
Andreas Czerniak	Bielefeld University	0000-0003-3883-4169							X
Anna Niehues	Radboud university medical center	0000-0002-9839-5439					X		
Anna-Lena Lamprecht	Utrecht University	0000-0003-1953-5606		X		X	X	X	X
Anne Claire Fouilloux	University of Oslo, Department of Geosciences	0000-0002-1784-2920					X		
Axel Loewe	Karlsruhe Institute of Technology (KIT)	0000-0002-2487-4744						X	X

Bala Desinghu	Rutgers University	0000-0003-2854-9583					X		
Ben van Werkhoven	Netherlands eScience Center	0000-0002-7508-3272						X	
Carlos Martinez	Netherlands eScience Center	0000-0001-5565-7577		X	X	X	X	X	X
Carole Goble	The University of Manchester	0000-0003-1219-2137				X	X		
Catherine Jones	Science and Technology Facilities Council (STFC)	0000-0002-5112-835X					X	X	
Céline Richard	CIRAD Centre de coopération internationale en recherche agronomique pour le développement	0000-0003-0854-2659					X		
Charles Gray	Newcastle University	0000-0002-9978-011X					X		
Chris Erdmann	American Geophysical Union	0000-0003-2554-180X					X		
Daniel Garijo	Information Sciences Institute, University of Southern California (USC)	0000-0003-0454-7145				X	X	X	
Daniel S. Katz	University of Illinois Urbana-Cham	0000-0001-5934-7525	X	X	X	X	X	X	X

	paign								
Daniel Nüst	Institute for Geoinformatics, Opening Reproducible Research,, University of Münster; de-RSE	0000-0002-0024-5046					X		
Daniele Tartarini	University of Sheffield	0000-0002-8913-0156						X	
Elena Ranguelova	Netherlands eScience Center	0000-0002-9834-1756						X	
Esther Plomp	Delft University of Technology, Faculty of Applied Sciences	0000-0003-3625-1357					X		X
Fotis E. Psomopoulos	Institute of Applied Biosciences (INAB|CERTH)	0000-0002-0222-4273		X	X	X	X	X	X
Francoise Genova	Centre de Donnees astronomiques de Strasbourg (CDS)	0000-0002-6318-5028				X			
Hartwig Anzt	University of Tennessee / Karlsruhe Institute of Technology	0000-0003-2177-952X					X		
Hugh Shanahan	Royal Holloway, University of London	0000-0003-1374-6015						X	

Ilian Todorov	UKRI, Science and Technology Facilities Council (STFC)	0000-0001-7275-1784					X	X	
James McNally	ICPSR University of Michigan	0000-0002-6807-4538						X	
Javier Moldon	IAA-CSIC	0000-0002-8079-7608					X		
Jen Harrow	ELIXIR	0000-0003-0338-3070			X	X	X	X	X
Jessica Burnett	US geological survey	0000-0002-0896-5099					X		
Joanna Leng	University of Leeds	0000-0001-9790-162X						X	
Julián Garrido-Sánchez	Instituto de Astrofísica de Andalucía IAA	0000-0002-6696-4772					X		
Khalid Belhajjame	PSL, Paris-Dauphine University	0000-0001-6938-0820					X		
Laurents Sesink	Leiden University Libraries	0000-0001-7880-5413					X		
Leyla Jael Castro	ZB MED - Information Center Life Sciences	0000-0003-3986-0510		X	X	X	X	X	X
Lorraine Hwang	UC Davis, CIG	0000-0002-1021-3101					X		
Maggie Hellström	Lund University, Sweden and ICOS Carbon	0000-0002-4154-2610					X		

	Portal								
Malin Sandström	International Neuroinformatics Coordinating Facility (INCF)	0000-0002-8464-2494				X		X	X
Manodeep Sinha	Swinburne University of Technology	0000-0002-4845-1228					X	X	
Marcos Roberto Tovani-Palone	University of São Paulo, Brazil; Modestum Ltd, United Kingdom	0000-0003-1149-2437						X	
Mark D. Wilkinson	Polytechnic University of Madrid	0000-0001-6960-357X					X		
Mateusz Kuzak	Netherlands eScience Center	0000-0003-0087-6021			X	X	X		
Mathieu Servillat	Observatoire de Paris	0000-0001-5443-4128						X	X
Matthias Liffers	Australian Research Data Commons	0000-0002-3639-2080					X		
Merc Fox	University of Arizona	0000-0002-0726-7301						X	
Michelle Barker	Research Software Alliance	0000-0002-3623-172X	X		X	X	X	X	X
Morane Gruenpeter	Software Heritage, INRIA	0000-0002-9777-5560		X	X	X	X	X	X
Nadica	University of	0000-0002-39							X

Miljković	Belgrade	33-6076							
Neil P. Chue Hong	Software Sustainability Institute / EPCC, University of Edinburgh	0000-0002-8876-7606	X	X	X	X	X	X	X
Nick Lynch	Curlew Research	0000-0002-8997-5298					X		X
Patricia Herterich	Digital Curation Centre	0000-0002-4542-9906						X	X
Paula Andrea Martinez	Research Software Alliance	0000-0002-8990-1985		X	X	X	X	X	X
Paula Martinez Lavanchy	TU Delft Library	0000-0003-1448-0917					X		
Qian Zhang	New Digital Research Infrastructure Organization (NDRIO), Canada	0000-0003-1549-7358				X	X		X
Sandra Gesing	University of Notre Dame	0000-0002-6051-0673				X			
Sarah Stevens	University of Wisconsin-Madison	0000-0002-7040-548X					X		
Sergio Martinez Cuesta	University of Cambridge and AstraZeneca	0000-0001-9806-2805					X		
Sharif Islam	Naturalis Biodiversity Center, Distributed	0000-0001-8050-0299						X	

	System for Scientific Collections (DiSSCo)								
Silvio Peroni	University of Bologna, OpenCitations	0000-0003-0530-4305					X		
Stian Soiland-Reyes	The University of Manchester	0000-0001-9842-9718					X		
Susanna-Assunta Sansone	University of Oxford	0000-0001-5306-5690							X
Tom Bakker	Netherlands eScience Center							X	
Tom Honeyman	Australian Research Data Commons	0000-0001-9448-4023		X		X	X	X	X
Tom Pollard	PhysioNet	0000-0002-5676-7898						X	
Tovo Rabemanantsoa	French National Institute for Agricultural Research (INRAE), Food and Environment	0000-0002-8362-9474					X		
Udayanto Dwi Atmojo	Aalto University	0000-0002-6865-0806						X	
Vanessa Sochat	Stanford University	0000-0002-4387-3819					X		
Yo Yehudi	Wellcome Trust	0000-0003-2705-1724					X		