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, Limor Peer, Maggie Hellström, Malin Sandström, Manodeep Sinha, Mateusz Kuzak, Mathieu Servillat, Michael Barton, 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, Clement Jonquet, Daniel Nüst, Daniele Tartarini, Elena Ranguelova, Hartwig Anzt, Ilian Todorov, James McNally, Javier Moldon, Jean-Christophe Souplet, Jessica Burnett, Joachim Wuttke, Joris van Eijnatten, Julián Garrido-Sánchez, Keith Russell, Khalid Belhajjame, Laurents Sesink, Lorraine Hwang, Marcos Roberto Tovani-Palone, Mark D. Wilkinson, Matthias Liffers, Merc Fox, Nadica Miljković, Nick Lynch, Nicola Soranzo, Paul Secular, Paula Martinez Lavanchy, Peter Hill, Rob van Nieuwpoort, Roberto Di Cosmo, Sandra Gesing, Sarah Stevens, Sergio Martinez Cuesta, Silvio Peroni, Stian Soiland-Reyes, Tek Raj Chhetri, Tom Bakker, Tovo Rabemanantsoa, Vanessa Sochat, Wilhelm Hasselbring, Yo Yehudi;
and the FAIR4RS WG

(*) Lead authors, with equal contributions

The author list and ordering was determined by the contributions each author made to the document. The full list of contributors and their contributions can be found in Appendix C - Contributor List.

Citation

Chue Hong, N. P., Katz, D. S., Barker, M., Lamprecht, A-L, Martinez, C., Psomopoulos, F. E., Harrow, J., Castro, L. J., Gruenpeter, M., Martinez, P. A., Honeyman, T., et al. (2022). FAIR Principles for Research Software version 1.0. (FAIR4RS Principles v1.0). *Research Data Alliance*. DOI: https://doi.org/10.15497/RDA00068

FAIR4RS WG: https://www.rd-alliance.org/groups/fair-research-software-fair4rs-wg

Abstract

To improve the sharing and reuse of research software, the FAIR for Research Software (FAIR4RS) Working Group has applied 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),* and includes explanatory text to aid adoption. It is an outcome of the FAIR for Research Software Working Group (FAIR4RS WG) based on community consultations that started in 2019.

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

Date	**Version Number**	**Description**	**Editor(s)**
15/3/2022	1.0	First release of principles○	Neil Chue Hong
9/6/2021	0.3	Draft for formal RDA 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

○: *The pre-1.0 drafts of the FAIR4RS Principles included sections describing the drafting process - these are now published separately.*

Table of Contents

Introduction

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 four *foundational principles*: being Findable, Accessible, Interoperable and Reusable. These four foundational principles, and 15 *guiding principles* that provide further detail, were published in *"The FAIR Guiding Principles for scientific data management and stewardship"* (Wilkinson et al., 2016, shortened to the *FAIR Guiding Principles* in the remainder of this document).

This document presents the first release of the *FAIR Principles for Research Software* (shortened to the *FAIR4RS Principles* in the remainder of this document). This work is an outcome of the FAIR4RS for Research Software Working Group (FAIR4RS WG) that was established in 2020 with the aim of developing community-endorsed FAIR principles for research software. The FAIR4RS Principles represent the consensus view of the research software community after extensive consultation (Katz et al., 2021, Chue Hong et al., 2021a).

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 (EC, 2018, EC & EOSC EB, 2020, EC, 2020, Gruenpeter et al., 2020). Community-produced outcomes on applying FAIR to research software produced before February 2020 can be found in the Software Source Code identification Interest Group's Wiki FAIR4Software reading resources[1]. Newer resources can be found in the FAIR4RS collection on Zenodo[2] and the literature review completed by the FAIR4RS Working Group (Chue Hong et al., 2021b, FAIR4RS WG, 2021).

The ultimate goal of FAIR is to increase the 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 guide software creators and owners on how to make their software FAIR. 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 led by members of the Research Software Alliance (ReSA), Future Of Research Communications and E-Scholarship (FORCE11), and the Research Data Alliance (RDA) communities. The FAIR4RS WG is a global and interdisciplinary community composed of ~250 members[3] who have an interest in the application of FAIR principles to research software and other research outputs. The members have diverse roles such as software users, software developers and maintainers, academics, policy makers, infrastructure support staff, and funders.

[1] https://www.rd-alliance.org/group/software-source-code-ig/wiki/fair4software-reading-materials

[2] https://zenodo.org/communities/fair4rs

[3] https://www.rd-alliance.org/node/69317/members

FAIR Principles for Research Software

The FAIR4RS WG define research software (Gruenpeter et al., 2021) as:

Research Software includes source code files, algorithms, scripts, computational workflows and executables that were created during the research process or for a research purpose. Software components (e.g., operating systems, libraries, dependencies, packages, scripts, etc.) that are used for research but were not created during or with a clear research intent should be considered software in research and not Research Software. This differentiation may vary between disciplines.

The FAIR4RS Principles are:

F: Software, and its associated metadata, is easy for both humans and machines to find.
F1. Software is assigned a globally unique and persistent identifier. ● F1.1. Components of the software representing levels of granularity are assigned distinct identifiers. ● F1.2. Different versions of the software are assigned distinct identifiers. F2. Software is described with rich metadata. F3. Metadata clearly and explicitly include the identifier of the software they describe. F4. Metadata are FAIR, searchable and indexable.
A: Software, and its metadata, is retrievable via standardized protocols.
A1. Software is retrievable by its identifier using a standardized communications protocol. ● A1.1. The protocol is open, free, and universally implementable. ● A1.2. The protocol allows for an authentication and authorization procedure, where necessary. A2. Metadata are accessible, even when the software is no longer available.
I: Software interoperates with other software by exchanging data and/or metadata, and/or through interaction via application programming interfaces (APIs), described through standards.
I1. Software reads, writes and exchanges data in a way that meets domain-relevant community standards. I2. Software includes qualified references to other objects.
R: Software is both usable (can be executed) and reusable (can be understood, modified, built upon, or incorporated into other software).
R1. Software is described with a plurality of accurate and relevant attributes. ● R1.1. Software is given a clear and accessible license. ● R1.2. Software is associated with detailed provenance. R2. Software includes qualified references to other software. R3. Software meets domain-relevant community standards.

Table 1: The FAIR Principles for Research Software

Software as source code is the preferred form for applying the FAIR4RS Principles. But software can occur in other forms which may be more useful for some communities or may simply be the only way in which the owner is willing to deliver them. These other forms include binaries, containers or software as a service, amongst others. Many of these alternate forms make sense for applying many but not all of the FAIR4RS Principles. For instance the internal workings of compiled binaries cannot be inspected or verified, but access to them may improve reuse by end-user researchers. The FAIR4RS Principles can be applied to any research software, regardless of the license.

In the remainder of this document, each of the FAIR4RS Principles is described in more detail. 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* elaborates on the principle, explains the meaning of specific keywords, discusses the ramifications of adopting the principle and cross-references interrelated principles.

A comparison of the evolution of these principles from the original *FAIR Guiding Principles for scientific data management and stewardship* (Wilkinson et al., 2016, with foundational principle text taken from GO FAIR, 2018) through 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 drafts developed by the FAIR4RS WG (Chue Hong et al., 2021a) to these published FAIR4RS Principles is provided in Appendix B.

As with the FAIR Guiding Principles, the FAIR4RS Principles are intended to be aspirational. 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 the applicability of the principles is the responsibility of the providers of the infrastructure they use to fulfill them, 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.

Findable

F: Software, and its associated metadata, is easy for both humans and machines to find.

For software to be findable, its associated metadata are readable by both humans and machines. Metadata should follow domain-relevant community standards.

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

A piece of software is given an identifier which is both globally unique (not used to identify any other object, even on a different system) and persistent (long-lasting, including its resolution - the ability to use it to get to the identified source). The use of globally unique and persistent identifiers enables 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 software granularity (the "level of detail being implemented") and software versions (the "changes between implementations") are addressed by F1.1 and F1.2. This principle also relates to enabling accessibility to software, specifically A1.

F1.1. Components of the software representing levels of granularity are assigned distinct identifiers.

The use of identifiers for more than the software project (often synonymous with "software concept" or "software product") improves findability by enabling components to be assigned distinct identifiers e.g, a software library, and a function in that library. The relationship between these components is embodied in the associated metadata. 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.

F1.2. Different versions of the software are assigned distinct identifiers.

To make different versions of the same software (or component) findable, each version needs to be assigned a different identifier. The relationship between versions is embodied in the associated metadata. What is considered a "version" is defined by the owner of the software: in many cases this will be something that the owner wants to specifically identify and use and/or "release" or "publish" so that others can use and reference/cite.

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.

Capturing the relationships between different versions of software will lead to greater understanding of the evolution of code, its authorship, ownership, description and purpose, amongst other things.

F2. Software is described with rich metadata.

Software requires descriptive metadata to support indexing, search and discoverability. This metadata must itself be FAIR (F4), 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. 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; see F4) and the software should be made explicit by mentioning the software's globally unique and persistent identifier in its associated metadata. In conjunction with A1, this means the metadata describes how the software can be obtained. Metadata are not required to include references for all of the softwares dependencies in order for software to be findable. I2 and R2 describe how references to dependencies increase the likelihood that software is interoperable and reusable.

F4. Metadata are FAIR, searchable and indexable.

Making the metadata about the software FAIR, including making it readable and discoverable by both humans and machines, improves 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 repository, or by a search engine. FAIR metadata also enables and encourages citation of research software.

Accessible

A: Software, and its metadata, is 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 or, in the case of software as a service, the software functionality is accessible via standardized protocols.*

In software engineering, "accessibility" can also be used to refer to the ability to access or use the software regardless of impairment, e.g. a disability. While this is important for research software, this is not how the original FAIR Guiding Principles define accessibility. It is recommended that the practice of making software usable by as many people as possible is best addressed in R3, and as separate guidance complementing 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 obtaining the software should not require specialized or proprietary tools or communication methods. For much software, there are commonly used technical communications protocols used to access the software, such as HTTPS.

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

It is the openness of the communications 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.

The FAIR Guiding Principles put specific emphasis on enhancing the ability of machines to use digital objects. In the context of software, there are often conditions of access, for instance, requiring a license server to be contacted, requirement for payment before use, or restrictions based on the privilege level of the user.

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, or because dependencies are no longer available. 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: Software interoperates with other software by exchanging data and/or metadata, and/or through interaction via application programming interfaces (APIs), described through standards.

The definitions of interoperability and reusability as defined by the FAIR Guiding Principles overlap when applied to software. To differentiate between the two, interoperability here is limited to being concerned with the capacity to exchange data between independent software (i.e., software that can be executed separately). As an example, the sense of "integrated" that applies to data (where two pieces of data combine to form new data) 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 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 data.

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 and metadata types, controlled vocabularies, and formats that are formally defined through standards to facilitate the exchange. Whereas F4 requires that metadata describing the software are FAIR, this principle ensures that the way that software interacts with other software is clearly described. A domain-relevant standard is an agreed standard that addresses the needs of a given community (or communities). 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, and they should be open APIs where possible.

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). Ideally, the data would be FAIR as well, and references to external data would be fully qualified. Qualified references should be to digital objects (e.g., metadata, other software, data), as well as to non-digital objects that have a virtual *presence in digital systems (e.g., samples, reagents, instruments, etc.) with which 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. Ideally this is in a form that includes a resolver within the reference (e.g., in the form of a persistent identifier, or URL). This information can also improve the reusability of software by explicitly including references to articles and data sets that document its use. Examples of qualified references might include: software X is implemented using software A (a programming language); software X uses software B (a library/dependency); software X is tested within software C (a platform); software X extends software D.*

Reusable

R: Software is both usable (can be executed) and reusable (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) here 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. The form in which it is made available changes the way it can be used. 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.

The concept of software quality overlaps with the FAIR4RS Principles, particularly reusability, but is not directly addressed by them. Similar to openness, software quality is beneficial to making software FAIR but not required.

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. References provided by applying I2 can help document the use and purpose of the software. 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.

Metadata and documentation are distinct, potentially complementary, concepts. Metadata about software can be included in documentation, or in the code itself, or in a separate location. Metadata included in the documentation are generally not machine readable, or indexable. This does, however, support the reusability of software particularly from a human perspective.

R1.1. Software is given a clear and accessible license.

To enable reuse, software must have a license that clearly describes how it can be used and reused, ideally with conditions that are readable by humans and machines. Licenses are often referred to by name, but machine readable licenses can be specified by reference to a standard vocabulary such as the SPDX License List[4] (SPDX Consortium, 2020). To support a wide range of reuse scenarios, the license should be as unrestrictive as possible and, to avoid license proliferation, choosing a widely used and recognized license is strongly recommended. 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.

Software provenance is a type of metadata that describes why and how the software came to be, as well as who contributed what, when and where. Provenance is sometimes referred to as lineage or pedigree. This extends beyond capturing a log of changes to source code as it is developed. Good provenance metadata clarifies the origins and intent behind the development of the software, and establishes authenticity and trust. As a type of metadata this overlaps with the metadata called for in guiding principles F2 and F4.

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 (e.g., requirements, imports, libraries) which are 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. In many programming languages, base methods or functions take a reference to a named entity, possibly in combination with a version number or qualifying domain and resolves this to a source. This principle goes beyond this in calling for qualified references to external dependencies, meaning that the reference itself resolves to the source via the qualifying authority.

[4] https://spdx.org/licenses/

This guiding principle calls for qualified references in software to other software (in aid of reuse). Principle I2 calls for qualified references to anything other than software (in aid of interoperability).

R3. Software meets domain-relevant community standards.

Software, including its documentation and license, should meet domain-relevant community standards and coding practices (e.g., choice of programming language, standards for testing, usage of file formats, accessibility [in the sense of usable by as many people as possible]) 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.

Communities can encompass research domains, programming languages, and technical approaches. Examples of community standards might include: BioSchemas from ELIXIR for describing resources in the life sciences and schema.org for general description of resources; Common Workflow Language; and the package managers commonly used by a programming language such as Maven (Java), npm (Javascript), PyPI (Python) and CRAN (R). It is important to note that the FAIR Guiding Principles address research outputs, not research processes, so standards should also be limited to best practice about the software itself, not the process of designing, developing, or maintaining it, such as the R community standards for creating packages[5] or the PEP 8 Style Guide for Python Code[6].

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.

[5] https://r-pkgs.org/description.html
[6] https://www.python.org/dev/peps/pep-0008/

References

Chue Hong, N. P., Katz, D. S., Barker, M., Lamprecht, A.-L., Martinez, C., Psomopoulos, F. E., Harrow, J., Castro, L. J., Gruenpeter, M., Martinez, P. A., Honeyman, T., et al. (2021). FAIR Principles for Research Software (FAIR4RS Principles). Research Data Alliance. https://doi.org/10.15497/RDA00065

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

European Commission. Directorate General for Research and Innovation. (2020). Scholarly infrastructures for research software: report from the EOSC Executive Board Working Group (WG) Architecture Task Force (TF) SIRS. Publications Office. https://doi.org/10.2777/28598

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.P., Martinez-Ortiz, C., Sesink, L., Liffers, M., Fouilloux, A.C., Erdmann, C., Peroni, S., Lavanchy, P.M., Todorov, I &

Sinha, M.. (2021). Defining Research Software: a controversial discussion (Version 1). https://doi.org/10.5281/zenodo.5504016.

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

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

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

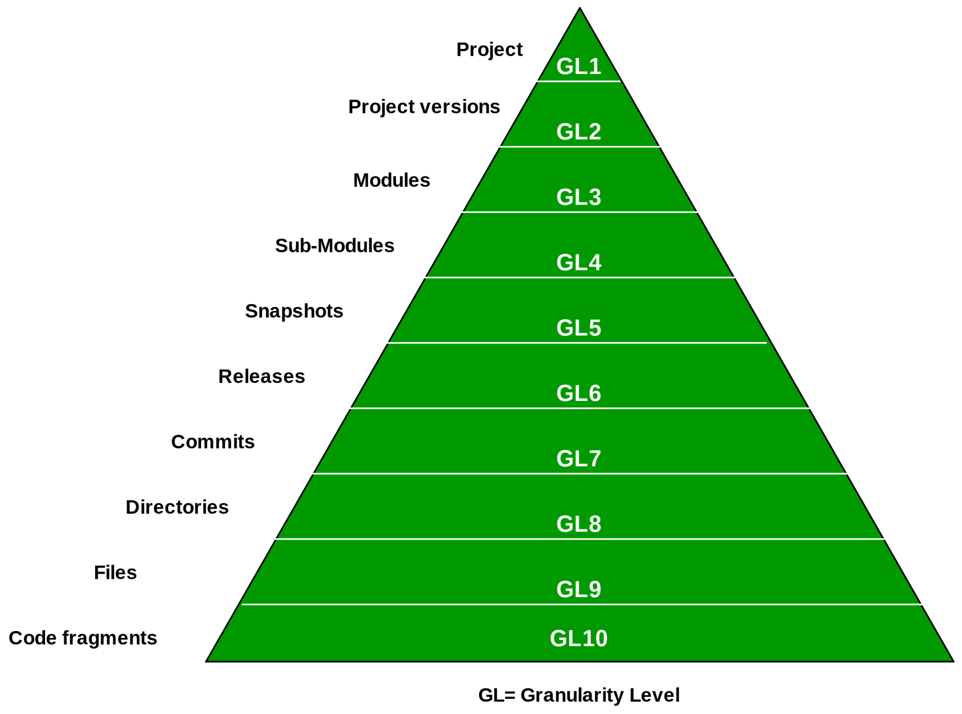

Figure 1: Granularity levels for software as identified by the RDA/FORCE11 Software Source Code Identifiers WG (RDA/FORCE11 SSCID WG et al., 2020)

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), the previous draft for community review (Chue Hong et al., 2021a) 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 Draft for RDA Community Review (2021)**	**FAIR4RS Principles (2022)**
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.	Software, and its associated metadata, is easy for both humans and machines to find.
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. Software is assigned a globally unique and persistent identifier.
			F1.1. Different components of the software must be assigned distinct	F1.1. Components of the software representing levels of granularity are assigned distinct

			identifiers representing different levels of granularity.	identifiers.
			F1.2. Different versions of the same software must be assigned distinct identifiers.	F1.2. Different versions of the software are 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.	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.	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.	F4. Metadata are FAIR, 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.	Software, and its metadata, is 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. 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.	A1.1. 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.	A1.2. The protocol allows for an authentication and authorization procedure, where necessary.
A2. Metadata are accessible, even when the data are no longer available	A2. Software metadata are accessible, even when the software is no longer available.	A2. Metadata are accessible, even when the software is no longer available	A2. Metadata are accessible, even when the software is no longer available.	A2. Metadata are accessible, even when the software is no 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).	Software interoperates with other software by exchanging data and/or metadata, and/or through interaction via application programming interfaces (APIs), described through standards.
I1. (Meta)data use a formal, accessible, shared, and broadly applicable language for	I1. Software and its associated metadata use a formal, accessible, shared and broadly	I1. Software should read, write or exchange data in a way that meets domain-relevant	I1. Software reads, writes and exchanges data in a way that meets domain-relevant	I1. Software reads, writes and exchanges data in a way that meets domain-relevant

knowledge representation.	applicable language to facilitate machine readability and data exchange.	community standards	community standards.	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.		*Now split between F4 and I1.*	*Now split between F4 and I1.*
	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.	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	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	The software is both usable (it can be executed) and reusable (it can be understood, modified, built upon, or incorporated into other software).	Software is both usable (can be executed) and reusable (can be understood, modified, built upon, or incorporated into other software).

different settings.		appropriately structured so that it can be replicated, combined, reinterpreted, reimplemented, and/or used in different settings.		
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. 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.	R1.1. Software is given 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.	R1.2. 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.	R3. Software meets domain-relevant community standards.
		R2. Software includes qualified references to other software	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.

Name	Institution	ORCID	Editor	Member of drafting group	Member of WG steering committee	Contributor to WG meetings	Contributor to subgroup reports (Jun-Dec 2020)	Contributor to first consultation (Feb-Mar 2021)	Contributor to second consultation (May 2021)	Contributor during RDA community review (Jun-Jul 2021)
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			
Clement Jonquet	INRIA	0000-0002-2404-1582								†

Daniel Garijo	Information Sciences Institute, University of Southern California (USC)	0000-0003-0454-7145				X	X	X		
Daniel S. Katz	University of Illinois Urbana-Champaign	0000-0001-5934-7525	X	X	X	X	X	X	X	X
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				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			
Jean-Christophe Souplet	CNRS	0000-0002-5112-2118								†
Jen Harrow	ELIXIR	0000-0003-0338-3070			X	X	X	X	X	
Jessica Burnett	US geological survey	0000-0002-0896-5099					X			
Joachim Wuttke	Forschungszentrum Jülich	0000-0002-4028-1447								X
Joanna Leng	University of Leeds	0000-0001-9790-162X						X		
Joris van Eijnatten	Netherlands eScience Center	0000-0002-8865-0002								X
Julián	Instituto de	0000-0002-6					X			

Garrido-Sánchez	Astrofísica de Andalucía IAA	696-4772								
Keith Russell	Australian Research Data Commons	0000-0001-5390-2719								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 Centre for Life Sciences	0000-0003-3986-0510		X	X	X	X	X	X	X
Limor Peer	Yale University	0000-0002-3234-1593							X	X
Lorraine Hwang	UC Davis, CIG	0000-0002-1021-3101					X			
Maggie Hellström	Lund University, Sweden and ICOS Carbon Portal	0000-0002-4154-2610					X			
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		
Michael Barton	Arizona State University	0000-0003-2561-1927								X
Michelle Barker	Research Software Alliance	0000-0002-3623-172X	X		X	X	X	X	X	X
Morane Gruenpeter	Software Heritage, INRIA	0000-0002-9777-5560		X	X	X	X	X	X	X †
Nadica Miljković	University of Belgrade	0000-0002-3933-6076							X	
Neil P. Chue Hong	Software Sustainability Institute / EPCC, University of Edinburgh	0000-0002-8876-7606	X	X	X	X	X	X	X	X

Nick Lynch	Curlew Research	0000-0002-8997-5298					X		X	
Nicola Soranzo	Earlham Institute	0000-0003-3627-5340								X
Patricia Herterich	Digital Curation Centre	0000-0002-4542-9906						X	X	
Paul Secular	University of Bath	0000-0002-4742-5351								X
Paula Andrea Martinez	Research Software Alliance	0000-0002-8990-1985		X	X	X	X	X	X	X
Paula Martinez Lavanchy	TU Delft Library	0000-0003-1448-0917					X			
Peter Hill	University of York	0000-0003-3092-1858								X
Qian Zhang	New Digital Research Infrastructure Organization (NDRIO), Canada	0000-0003-1549-7358				X	X		X	
Rob van Nieuwpoort	Netherlands eScience Center	0000-0002-2947-9444								X
Roberto Di Cosmo	Software Heritage, INRIA, Université Paris Diderot	0000-0002-7493-5349								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 System for Scientific Collections (DiSSCo)	0000-0001-8050-0299						X		
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	
Tek Raj Chhetri	University of Innsbruck	0000-0002-3905-7878								X
Tom Bakker	Netherlands eScience Center							X		
Tom Honeyman	Australian Research Data Commons	0000-0001-9448-4023		X		X	X	X	X	X
Tom Pollard	PhysioNet	0000-0002-5676-7898						X		
Tovo Rabemanantsoa	French National Institute for Agricultural Research	0000-0002-8362-9474					X			

	(INRAE), Food and Environment									
Udayanto Dwi Atmojo	Aalto University	0000-0002-6865-0806						X		
Vanessa Sochat	Stanford University	0000-0002-4387-3819					X			
Wilhelm Hasselbring	Christian-Albrechts-Universität zu Kiel	0000-0001-6625-4335								X
Yo Yehudi	Wellcome Trust	0000-0003-2705-1724					X			X

† RDA Atelier participant