university of groningen

Digital Competence Centre

How to FAIRify Your Research Software? A Practical Overview

What is FAIR software?

The FAIR Guiding Principles promote the reuse of research data by making it more findable, accessible, interoperable and reusable by humans and machines.
These principles are meant to not only apply to data, but also to other digital objects (e.g. algorithms, tools, and workflows) that enable to conduct research, as all the components of research must be available to ensure transparency, reproducibility and reusability. In fact, the European Commission expert group on FAIR data also states the following: "Central to the realization of FAIR are FAIR Digital Objects, which may represent data, software or other research resources."
Technically, software is represented by a sequence of ones and zeros that are interpreted as computer instructions. This puts data and software in the same category of digital data, as they are written in the same way (ones and zeros), only the former is meant to be read, while the latter is meant to be executed.
In the more abstract context of FAIR, software and data are regarded as different kinds of digital research objects next to each other. They both share particular characteristics that allow them to be treated alike for certain aspects of FAIR, such as the possibility of having a Digital Object Identifier (DOI) assigned or having a license, and other characteristics that require the FAIR principles as applied to data to be adapted and expanded upon when treating software.
This overview will both provide a short comparison between data and software, illustrate the FAIR principles applied to software, as well as a guide on how to ensure that your software follows said FAIR principles.
(Mention data autonomy?)

Your one-stop for research IT and data

Important differences between research data and research software

As stated above, research software constitutes any piece of executable code that contributes to collecting, analyzing or in any way modifying the data attached to your research. So, what aspects of data and software make them fundamentally different from each other?

- Software is a tool to perform a specific task (e.g. processing data), data provides facts and observations.
- Software is a subjective way of processing and interpreting data and can change over time with new discoveries, data is static and objective evidence collected during research.
- Software is executable, data is not.
- Software depends on other software to operate (such as compilers and editors, for example), data is independent of other data.
- The lifetime of software is generally shorter than that of data, as versioning is applied more frequently.

The FAIR principles applied to research software

Here we will give a short summary and overview of what the four FAIR principles look like when applied to research software, as presented in Lamprecht et al. (2019). Specific attention to where to find resources and how to realize these principles will be given in the “How to FAIRify your software” section down below.

1) Findability

This is the point that changes the least compared to research data. As for data, software should be made findable by following the steps listed below:

- ➢ Software and its associated metadata have a global, unique and persistent identifier for each released version.
- ➢ Software is described with rich metadata.
- ➢ Metadata clearly and explicitly include identifiers for all the versions of the software it describes.
- ➢ Software and its associated metadata are included in a searchable software registry.

2) Accessibility

As for point 1 above, this point also follows the FAIR principle laid down for research data closely, with only slight alterations to better describe the reality of software:

- ➢ Software and its associated metadata are accessible by their identifier using a standardized communications protocol.
- ➢ The protocol is open, free, and universally implementable.
- ➢ The protocol allows for an authentication and authorization procedure, where necessary.
- ➢ Software metadata is accessible, even when the software is no longer available.

3) Interoperability

This is probably the point that differs the most compared to data. While data prescribes informal formats to allow a variety of systems and programs to read it, software cannot be informal by its very nature. As such, the FAIR principle has been adapted and expanded in this way:

- ➢ Software and its associated metadata use a formal, accessible, shared and broadly applicable programming language to facilitate machine readability and data exchange.
- ➢ Software and its associated metadata are formally described using controlled vocabularies that follow the FAIR principles.
- ➢ Software use and produce data in types and formats that are formally described using controlled vocabularies that follow the FAIR principles.
- ➢ Software dependencies are documented and mechanisms to access them exist.

4) Reusability

Lastly, this point has also been adapted to software in the following way:

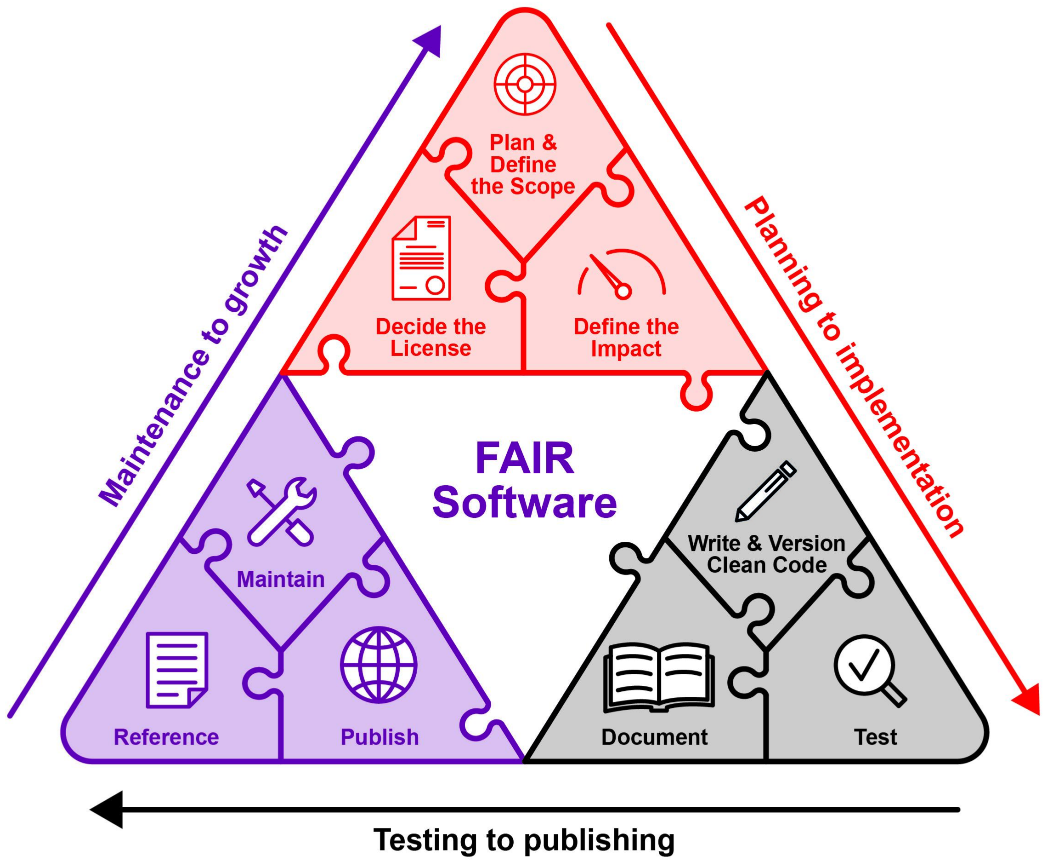
- ➢ Software and its associated metadata are richly described with a plurality of accurate and relevant attributes.
- ➢ Software and its associated metadata have independent, clear and accessible usage licenses compatible with the software dependencies.
- ➢ Software metadata include detailed provenance, detail level should be community agreed.
- ➢ Software metadata and documentation meet domain-relevant community standards.

Why FAIR Software?

- Modern research relies heavily on the availability and development of software.
- Findable and accessible software creates recognition for the author.
- Interoperable and reusable software enables a faster growth of the knowledge in the community.
- FAIR software benefits from having a given research process cited, tested, and validated by the community.
- Using a software repository to make your code FAIR allows you to keep track of the versions of your software.
- FAIR software enables transparency, reproducibility, and reusability in research.
- FAIR software allows science, education, society, and industry to have effective access to software-based knowledge.
- FAIR software should be developed with the handling and creation of FAIR data in mind.

How to FAIRify your Software

Developing FAIR software is easiest if it is created as FAIR from the ground up. This does not mean that you cannot make existing software FAIR, but rather that the FAIR principles are more effective if they are integrated in the very essence of your work from the beginning. This document will provide a quick guide on what to look for when making software FAIR, but please keep in mind that the DCC can and will still help you along the way. So, without further ado, what do you need to make your software FAIR and how do you get it?



Software management should cover three main points:

- ➢ detail the **purpose** of your software,
- ➢ make sure it is and **stays reliable** throughout your research and for as long as it is in use,
- ➢ make sure that it is kept **maintained** after its initial release.

Purpose Phase

- Compile a Software Management Plan (SMP) to help you plan for important aspects of your software.
    - You can read more on Research Software Management Plan on the DCC website.
- Think about and plan the (initial) scope of your software.
    - The scope you define here does not need to be final, but it will give you an idea as to what you need to develop for the first version of your software.
- Define the impact your software will have, based on the scope you defined in the previous step.
    - For an overview of how impactful your software can be, please read the Research Software Categories page on the DCC website.
- Decide on the license you wish to distribute the software under.
    - For more information on software licenses you can read this compact guide on the DCC website: Research Software License.
    - You can also find a list of licenses in alphabetical order here: Licenses by Name | Open Source Initiative,
    - and a short decision tree to help you Choose a License.
    - If you are in doubt about which license you want to use and for questions about potential commercial use, please consult the University Services: IP & Business Development | University Services.

Coding & Reliability Phase

- Choose a searchable platform with version control to put your software on, in order to make it findable and accessible.
    - Examples of platforms you can use for making your software FAIR are:
        - GitLab (managed by your institution)
        - GitHub (externally managed)
        - Or you could use a publicly accessible repository with version control
- Write and version clean code on the platform you chose.
- When working on your code, make sure to properly comment it. Also create clear code documentation in the form of a manual for example.
    - Here is some inspiration on how to comment your code: Best practices for writing code comments - Stack Overflow Blog

  - And here is a short blog post on how to approach writing a user-focused manual: How To Write Manuals as a Developer | by Andreas D. | Better Programming
- Describe your code with rich metadata.
  - For more information on software metadata please follow this link: Research Software Metadata
- Properly test your code during each step of development.
- Finally, you will need to make sure that the data you produce is also FAIR. Please review our FAIR data & Open Science guide or any other source on the topic to make sure that your output data fits the four FAIR data pillars.

Maintenance Phase

- Publish your code in a Findable and Accessible repository, like Zenodo.
  - Obtain a **DOI for each version** of the software that you publish.
  - If you are using GitHub, you can make use of the GitHub integration offered as a functionality by **Zenodo** (see example below).
- Make sure that the license and your citation file are **correct and up-to-date**.
  - The scope of your software may have changed from the original idea, so double-checking this at the end of a development cycle is advised.
  - Here is a short video on how to create a citation file: Software Citation: how to do it?
- Maintain your code, fix reported bugs, and verify the functionality of your dependencies!
  - If you want to respect the FAIR software principles, it is important that your code remains reliable after your research is done.
- **Plan for the next cycle!** Does your code need to be enlarged, improved, or have its scope redefined?
  - Review the SMP and start a new development cycle from the beginning of this guide, if your software requires new features.

FAIR Example

A researcher that has decided to use GitHub as their software repository, can then use their GitHub account to connect to Zenodo. This allows them to publish their software project in a FAIR way to Zenodo, not only to make it more findable by having it on another searchable platform, but also because Zenodo will then provide a persistent DOI for each published version. This DOI can then be made available in the citation file to both keep track of the version used and to simplify the discovery of the correct or current version of the program by other interested parties.
Here is a short guide on how to reference and cite content from GitHub to Zenodo.
Here is a concrete example: Noodles

References:

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. (2019). Towards FAIR principles for research software. *Data Science*, *3*(1), 37–59. https://doi.org/10.3233/ds-190026

Martinez-Ortiz, Carlos, Martinez Lavanchy, Paula, Sesink, Laurents, Olivier, Brett G., Meakin, James, de Jong, Maaike, & Cruz, Maria. (2022). Practical guide to Software Management Plans (1.0). Zenodo. https://doi.org/10.5281/zenodo.7248877

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