FAIR 4 Research Software (FAIR4RS)
RDA France

Morane Gruenpeter (Software Heritage, Inria - FAIRsFAIR)
Carlos Martinez, Neil Chue Hong, Daniel S. Katz, Paula A. Martinez, Michelle Barker,
Leyla Jael Castro, Jennifer Harrow, Fotis Psomopoulos

research data sharing without barriers
rd-alliance.org

15th October, 2021
Today’s goals

● Introduction
  ○ Research software
  ○ Present the FAIR4RS WG activities
● Review FAIR principles for research software
● Next steps
Multiple facets, it can be seen as:

- a **tool**
- a research **outcome** or result
- **the object** of research
Collect, preserve and share all software source code

Preserving our heritage, enabling better software and better science for all

Source: Software Heritage (June 2021)

https://archive.softwareheritage.org/
Why are we here? A plurality of needs

Researchers

- **archive and reference** software used and created in articles
- **find** useful software
- **get credit** for developed software
- **verify/reproduce/improve** results

Laboratories/teams

- **track** software contributions
- **produce** reports
- **maintain** web page

Research Organization

know its **software assets** for:

- technology transfer,
- impact metrics,
- strategy
L’ouverture des codes sources des logiciels est un enjeu majeur de reproductibilité des résultats scientifiques.
EOSC Scholarly Infrastructures for RS (SIRS)

Important policy tool in Open Science
- 9 infrastructures
  - 3 archives
  - 3 open access publishers
  - 3 aggregators
- recommendations
  - archive in Software Heritage,
  - use SWHID
  - open non profit
  - default to open source for research software

"all research software should be made available under an Open Source license by default, and all deviations from this default practice should be properly motivated"

See https://doi.org/10.2777/28598

Four pillars: Archive, Reference, Describe, Credit

EOSC Scholarly Infrastructures for Research Software

- Chairs
  - Roberto Di Cosmo, Software Heritage, Inria
  - José Benito Gonzalez Lopez, Zenodo, CERN
What is at stake? In order of difficulty

Archive
➔ Research software artifacts must be properly archived
➔ make it sure we can retrieve them (reproducibility)

Reference
➔ Research software artifacts must be properly referenced
➔ make it sure we can identify them (reproducibility)

Describe
➔ Research software artifacts must be properly described
➔ make it easy to discover them (visibility)

Cite (for credit)
➔ Research software artifacts must be properly cited (not the same as referenced!)
➔ to give credit to authors (evaluation!)
Software in RDA

Birds of a Feathers RDA P9, Barcelona, April 2017

● Software Source code: Sharing, Preservation and Reproducibility
● 60 participants

Software groups:

○ RDA, ReSA and FORCE11 FAIR for Research Software WG | 3.11 | 14:00 - 15:30 UTC | Breakout 2 session
○ RDA Curating for FAIR and Reproducible Research (CURE FAIR WG) | 3.11 | 22:30 - 00:00 UTC | Breakout 3 session
○ RDA Software Source Code IG (SSC IG) | 9.11 | 16:00 - 17:30 UTC | Breakout 8 Source code and reproducibility

3.11 | 22:30 - 00:00 | Breakout 3 BoF - Skills and training curriculums to support FAIR for Research Software

11.11 | 15:00 - 16:00 UTC | Plenary session: Data and Software Sharing: The Role of Societies and Associations
Software is not just another type of data

**Recommendation n°5:**
Recognise that FAIR guidelines will require *translation for other digital objects* and support such efforts.

2019: ‘Six Recommendations for Implementation of FAIR Practice’
(*FAIR Practice TF, 2020*)

**Recommendation n° 2:**
Make sure *the specific nature of software* is recognized and not considered as “just data” particularly in the context of discussion about the notion of FAIR data.

2019: the *Opportunity Note* by the French national Committee for Open Science's Free Software and Open Source Project Group
(*Clément-Fontaine, 2019*)
Software in the FAIR ecosystem

Webinar FAIR + Software: decoding the principles (Nov 2020)

October 16, 2020

M2.15 Assessment report on ‘FAIRness of software’

Software has an important place in academia and as such, it has an important place in the FAIR ecosystem. Software can be used throughout the research process; however, it can also be an outcome of the research process. Distinguishing between these different roles is essential for any assessment of the ‘FAIRness of software’.

This is the first milestone of the FAIR4FAIR project focused specifically on software as a digital object. In this report, we discuss the state-of-the-art of software in the scholarly ecosystem in general and in the FAIR literature in particular. We identify the challenges of different stakeholders when it comes to finding and reusing software. Furthermore, we present an analysis of nine resources that call for the recognition of software in academia and that present guidelines or recommendations to improve its status—either by becoming more FAIR or by improving the curation of software in general. With this analysis, we demonstrate how wide each of the FAIR principles is seen as relevant, achievable, and measurable, and in what sense, it benefits software artifacts. Finally, we present 10 high-level recommendations for organizations that seek to define FAIR principles or other requirements for research software in the scholarly domain.

Feedback and suggestions will be most welcome as comments on the public Google Doc version of this report:
https://docs.google.com/document/d/1yxidvLhka3KxshJq4C0hX7aHlKkxwC8BdwfMwhuVYB6Q/edit#v=sharing

Ecosystem components, to highlight the software roles in the Ecosystem, the symbol </> was added (Original diagram 3 from L’Hours & Von Stein, 2020)
FAIR enablement:
Enable / Respect / Reduce
Enable: Augment / Facilitate

Technically-oriented
- FAIR enablement
- Quality of service
- Open & Connected

Socially-oriented
- User centricity
- Transparency
- Longevity
- Ethical & Legal

D2.7 Framework for assessing FAIR Services
https://doi.org/10.5281/zenodo.5336233
Main objective

Defining FAIR principles for research software

Steering committee and WG chairs:

Morane Gruenpeter, Paula A. Martinez, Carlos Martinez, Michelle Barker, Daniel S. Katz, Leyla Garcia, Neil Chue Hong, Fotis Psomopoulos and Jennifer Harrow

- Acknowledging the ~228 members and contributors of the FAIR for Research Software working group #FAIR4RS
FAIR4RS initial subgroup activities and outputs

- 1. **A fresh look at FAIR for Research Software** examined the FAIR principles in the context of research software from scratch, not based on pre-existing work. **Lead: Daniel S. Katz**

- 2. **FAIR work in other contexts** examined efforts to apply FAIR principles to different forms including workflows, notebooks and training material, to provide insights for the definition and implementation of FAIR principles for research software. **Lead: Michelle Barker**

- 3. **Defining Research Software: a controversial discussion** reviews existing definitions of research software in order to provide the overall context of the subgroup outputs. **Lead: Morane Gruenpeter**

- 4. **Review of new research related to FAIR Software** reviewed new research around FAIR software that has come out since the release of the *Towards FAIR principles for research software* (Lamprecht et al., 2019). **Lead: Neil Chue Hong**
Defining the scope

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 minimal requirement for achieving computational reproducibility is that all the computational components (Research Software, software used in research, documentation and hardware) used during the research are identified, described, and made accessible to the extent that is possible.

Research Software ≠ Software in research
First community consultation to get feedback on findings of subgroups, and questions around scope of the draft FAIR4RS principles

- 24 February - 10 March 2021
- 215 comments from 19 named contributors (other than the SC) + other anonymous contributors.

Used as main input for FAIR4RS drafting sprints, with questions and clarifications raised by the community discussed by the drafting team to determining the intent of the principles.
Draft for formal community review of the draft FAIR4RS principles

- **Review period:** Friday, 11 June, 2021 to Sunday, 11 July, 2021

During this period the WG actively engaged with the community to respond to their feedback and capture their comments in the best possible way.

Drafting group meet in August to address comments received.
Community engagement

60+ events

Locations:

Channels:
Mailing list, Webinars, Twitter, RDA website, GitHub, etc.

Events:
CarpentryCon, SORSE, IEEE eScience, CW21, Collegeville, Utrecht University, PIDapalooza, INCF, etc.

July 2020

Dec 2021

Development of the FAIR4RS Principles

- Intent and methods of the FAIR Guiding Principles taken as starting point:
  - “maximize the added-value gained by contemporary, formal scholarly digital publishing”
  - “to ensure transparency, reproducibility, and reusability.”

- The FAIR Principles are aspirational, and FAIR is not binary
  - The aim of FAIR (and FAIR) metrics is to show progress to increasing FAIRness

- Software encompasses many forms, which may benefit different users
  - Source code is often the most useful form to understand the software, and the easiest form to apply the FAIR4RS Principles.

- Many software engineering practices are relevant to the FAIR4RS Principles
  - For instance: localization can improve findability, 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.
**Findable:** Software, and its associated metadata, is easy to find for both humans and machines.

- **F1.** Software is assigned a globally unique and persistent identifier
  - F1.1. Different components of the software are assigned distinct identifiers representing different levels of granularity
  - F1.2. Different versions of the same 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 and are searchable and indexable

**Interoperable:** Software interoperates with other software through 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

**Reusable:** Software is both usable (it can be executed) and reusable (it 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>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>F. Findable</strong></td>
<td><strong>Software, and its associated metadata, is easy to find for both humans and machines.</strong></td>
</tr>
<tr>
<td>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.</td>
<td></td>
</tr>
<tr>
<td><strong>F1. (Meta)data are assigned a globally unique and persistent identifier</strong></td>
<td><strong>F1. Software is assigned a globally unique and persistent identifier.</strong></td>
</tr>
<tr>
<td></td>
<td><strong>F1.1. Different components of the software are assigned distinct identifiers representing different levels of granularity.</strong></td>
</tr>
<tr>
<td></td>
<td><strong>F1.2. Different versions of the same software are assigned distinct identifiers.</strong></td>
</tr>
<tr>
<td><strong>F2. Data are described with rich metadata (defined by R1 below)</strong></td>
<td><strong>F2. Software is described with rich metadata.</strong></td>
</tr>
<tr>
<td></td>
<td><strong>F3. Metadata clearly and explicitly include the identifier of the data they describe</strong></td>
</tr>
<tr>
<td></td>
<td><strong>F4. Metadata are FAIR and is searchable and indexable.</strong></td>
</tr>
<tr>
<td><strong>F4. (Meta)data are registered or indexed in a searchable resource</strong></td>
<td></td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td><strong>A. Accessible</strong></td>
<td></td>
</tr>
<tr>
<td>Once the user finds the required data, she/he needs to know how can they be accessed, possibly including authentication and authorisation.</td>
<td>Software, and its metadata, is retrievable via standardized protocols.</td>
</tr>
<tr>
<td>A1. (Meta)data are retrievable by their identifier using a standardized communications protocol</td>
<td>A1. Software is retrievable by its identifier using a standardized communications protocol.</td>
</tr>
<tr>
<td>A1.1. The protocol is open, free, and universally implementable</td>
<td>A1.1. The protocol is open, free, and universally implementable.</td>
</tr>
<tr>
<td>A1.2. The protocol allows for an authentication and authorization procedure, where necessary</td>
<td>A1.2. The protocol allows for an authentication and authorization procedure, where necessary.</td>
</tr>
<tr>
<td>A2. Metadata are accessible, even when the data are no longer available</td>
<td>A2. Metadata are accessible, even when the software is no longer available.</td>
</tr>
</tbody>
</table>
Interoperable

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>I. Interoperable</td>
<td></td>
</tr>
<tr>
<td>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.</td>
<td>Software interoperates with other software through exchanging data and/or metadata, and/or through interaction via application programming interfaces (APIs), described through standards.</td>
</tr>
<tr>
<td>I1. (Meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation.</td>
<td>I1. Software reads, writes and exchanges data in a way that meets domain-relevant community standards.</td>
</tr>
<tr>
<td>I2. (Meta)data use vocabularies that follow FAIR principles</td>
<td></td>
</tr>
<tr>
<td>I3. (Meta)data include qualified references to other (meta)data</td>
<td>I2. Software includes qualified references to other objects.</td>
</tr>
</tbody>
</table>
# Reusable

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>R. Reusable</strong></td>
<td></td>
</tr>
<tr>
<td><strong>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.</strong></td>
<td><strong>Software is both usable (it can be executed) and reusable (it can be understood, modified, built upon, or incorporated into other software).</strong></td>
</tr>
<tr>
<td><strong>R1. (Meta)data are richly described with a plurality of accurate and relevant attributes</strong></td>
<td><strong>R1. Software is described with a plurality of accurate and relevant attributes.</strong></td>
</tr>
<tr>
<td>R1.1. (Meta)data are released with a clear and accessible <strong>data usage license</strong></td>
<td><strong>R1.1. Software is given a clear and accessible license.</strong></td>
</tr>
<tr>
<td>R1.2. (Meta)data are associated with detailed provenance</td>
<td><strong>R1.2. Software is associated with detailed provenance.</strong></td>
</tr>
<tr>
<td>R1.3. (Meta)data meet domain-relevant community standards</td>
<td><strong>R3. Software meets domain-relevant community standards.</strong></td>
</tr>
<tr>
<td><strong>R2. Software includes qualified references to other software.</strong></td>
<td></td>
</tr>
</tbody>
</table>
Who is responsible for FAIR software?

Who is expected to apply FAIR?

- And why?

“...the application of the FAIR4RS Principles is the responsibility of the owners (who are often the creators) of the software, not the users. “

“The FAIR4RS Principles are also relevant to the larger ecosystem and various stakeholders that support research software (e.g., repositories and registries).”

Adapted by Neil Chue Hong from original by Brian Nosek: Strategy for Culture Change (2019)
Beyond FAIR: FAIR is not the end goal

(Katz et al. 2020) arXiv:2101.10883
New subgroups

Three new subgroups now starting

- 5. Adoption guidelines
  - Identify, create, review existing resources that facilitate the adoption of FAIR4RS principles

- 6. Adoption support
  - Identify & start to work with organisations following FAIR4RS guidelines (or willing to do so)
  - Stimulate adoption of FAIR4RS guidelines
  - Document & share examples of FAIR4RS adoption (and plans)

- 7. Governance
  - Create communications plan and content that clarifies post-release governance structure

Join one or multiple subgroups via the form
Next Steps

● Publicize principles
● Via new subgroups:
  ○ Develop curriculum and training
  ○ Develop and encourage tooling to support applying principles
  ○ Encourage adoption, and highlight successful examples
  ○ Consider future governance of community and change processes

● Community work around gaps that prevent adoption
● Define metrics to measure adoption
  ○ For specific software
  ○ For principles and scholarly community as a whole
Get involved!

- Join the RDA group and be part of the mailing list
- Join one or multiple subgroups via the form
- Come to events
- Follow the steering committee meeting minutes
- Say ‘Hi’ on the gitter channel
- Visit and read the publications on Zenodo
- Review the bibliography collected on Zotero

All this information is detailed on the community engagement channels page
Thank you!

Questions?

Keep in touch: morane@softwareheritage.org
@moraneottilia, @SWHeritage
https://www.fairsfair.eu/fairsfair-newletters/
https://www.softwareheritage.org/newsletter/