

RS Workshop Session A

Descriptive metadata - curation tools and workflows

Links

- [CodeMeta repository](#) (vocabulary and crosswalk)
- [Issue-tracker](#)
- CodeMeta generator [hosted version](#)
 - Contributions are welcome on the [code repository](#)




🎯 Description & goals

- Discussion how **metadata curation** can support Research Software
- The case of the [HAL](#) deposit and the moderation process: “[Create software deposit in HAL](#)” ; [open science tutorials “Source code deposit”](#)
- **Group activity:** *Collect use cases and best practices to identify what metadata properties are needed*
- Discuss improvements of existing tools (e.g: [codemeta-generator](#)) and guidelines

Motivations: Master research software metadata

- Why do we need metadata for software
- Where is software metadata available
- Who is responsible for the metadata
- How can we provide/get metadata
- Using CodeMeta: vocabulary, crosswalk, tools

How will we work during this session?

<i>Time</i>	<i>Activity</i>	<i>On site referent</i>	<i>Online referent</i>	<i>Duration</i>
9.45-9.55	 Sub-group introductions and instructions for activity (exercises 1 and 2) <ul style="list-style-type: none"> - Choose your group - Choose rapporteur 	<i>Morane</i>	<i>Paula</i>	10'
9.55-10.20	Exercise 1: metadata property analysis			25'
	Exercise 2: identify use cases from the property analysis			
10.20-10.30	 Session A - mid session checkpoint, Q&A (on site and online) Review what we have and what to do next (with online group) <ul style="list-style-type: none"> - If remaining properties needs analysis Go back to ex. 1 and ex. 2			10'
10.30-10.45	 Exercises 1 and 2: last steps, get ready for the report back!			15'
10.45-11.15	Break			30'
11.15-11.30	Report back for session A and B in the live notes			Plenary

Activities

Exercise 1: metadata property analysis

Step A: Choose a group

Step B: Click on the name of the property to fill the dedicated template

Step C: Fill the template asking you: Why? Who? Where? How?

Step D: If your small group encounters complexities with the term, capture the difficulties in the "Discussion" section

Step E: If you agree the term or description should be modified in CodeMeta, open an issue here: <https://github.com/codemeta/codemeta/issues>

Groups (tribute to Sweden) 🇸🇪 🎤 🐻

Onsite groups

	Group Mamma Mia	Group Fernando	Group Winner takes it all	Group Waterloo
Properties	<ul style="list-style-type: none"> author funder affiliation Maintainer supportingData 	<ul style="list-style-type: none"> dateCreated dateModified datePublished referencePublication relatedLink 	<ul style="list-style-type: none"> Version runtimePlatform 	<ul style="list-style-type: none"> identifier url

Online Groups

	Group Gimme!	Group S.O.S	Group Dancing Queen

Properties	<ul style="list-style-type: none">• Description• Name• Readme (+documentation) + buildInstructions	<ul style="list-style-type: none">• keywords• programmingLanguage• operatingSystem• softwareRequirements	<ul style="list-style-type: none">• License• developmentStatus• embargoDate
------------	--	---	---

Exercise 2: Identify use cases from the property analysis

Part of exercise 1 is to answer the question “why do we need the property”. Go back to your answers and copy it to the “Use case collection” section on this document: [Why are we here? A collection of use cases.](#)

Bonus: complete the full scenario column.

Why are we here? A collection of use cases both online and offline

	Actor - Who?	Action - What?	Reason - Why?	Scenario
<i>metadata</i>	<i>Stakeholder that does the action</i>	<i>Action needed</i>	<i>The goal of the action, why do this actor needs this action</i>	As a [actor] I can [action] so that [reason]
CodeRepository	Software developer	open an issue	contribute to an existing tool to improve it	
identifier	Sw author or responsible asks for PID PID provider mints/calculates a PID	Ask Assign/Calculate	To make the resource easily identifiable to the world	As an Author/responsible of the software, I ask for a PID (extrinsic) or enable automatic calculation (intrinsic, e.g., by having the software in SWH) so my sw has a unique PID
author	Curator or from code repository/README	Entered by a curator or harvested from another site or within the software	attribution/contact	As a collaborator, I can identify a creator of the code so that they get attribution in a citation
dateModified	Developer / researcher			As a developer / researcher, I can check the

	Actor - Who?	Action - What?	Reason - Why?	Scenario
<i>metadata</i>	<i>Stakeholder that does the action</i>	<i>Action needed</i>	<i>The goal of the action, why do this actor needs this action</i>	As a [actor] I can [action] so that [reason]
				modified date so that I know if the software is “up to date” (has not been worked on a long time ago).
<i>BuildInstructions</i>	A user	<i>needs to install or run the software</i>	<i>To use the software locally or in its own hardware (such as a HPC, or cloud)</i>	
DateCreated	Developer of the software			As a developer I can have credit and attribution for a software I worked on.
Readme	A user	Wants to understand if the software is useful for their purpose	To select the software they are going to use	
Readme	A user	Wants to understand the requirements of using a given software (see also software requirements)	To decide whether or not the requirements can be met before running the software	
Readme	A user	Wants to know who to acknowledge	To add a reference to the software tool in a publication, or a poster or any other scholarly output	
datePublished	Developer / Aggregator		Citation	So that people know when it was released.

	Actor - Who?	Action - What?	Reason - Why?	Scenario
<i>metadata</i>	<i>Stakeholder that does the action</i>	<i>Action needed</i>	<i>The goal of the action, why do this actor needs this action</i>	As a [actor] I can [action] so that [reason]
version	User	refer a specific version of software	To be able to cite it for reproduction of older results from colleagues	As a user I can refer to a specific version of a software so that I can cite it when reproducing older research.
<i>Keyword</i>	<i>Search engine</i>	<i>Help in findability of the resource, but also in the classification of it. aries/Ontologies</i>	<i>Resources need to be classified in order to be easily findable</i>	A postdoctorant looking for some software helping in analysing his data
ProgrammingLanguage	User / search engine	search for language specific tools	To enable interoperability with other tools written in the same language To allow reuse by users who can use that language	A teacher wanting to illustrate a way to solve a problem using a specific language
runtimePlatform	User / Contributor	Use (or develop) the software	To have compatible environment where the software should work as intended	As a user (or contributor) I can use (or develop) the software so that I have a compatible environment where the software works as it should.
license	Authors of the software	Make a decision about a proper license and add (a) LICENSE file(s)	To make a clear statement, what are the conditions and terms that apply to use the product/software	As a developer I want to be able to clearly state, who should be allowed to do what with my software



	Actor - Who?	Action - What?	Reason - Why?	Scenario
<i>metadata</i>	<i>Stakeholder that does the action</i>	<i>Action needed</i>	<i>The goal of the action, why do this actor needs this action</i>	As a [actor] I can [action] so that [reason]
				As a user, I want to know, what I am allowed to do with this software (use it, build on it)
developmentStatus	Developers	Description of development status, e.g. Active, inactive, suspended. See repostatus.org	Inform the public if a software is live or outdated. Important information to decide, if I - as a researcher - want to use or build on this software for my research	As a user, I want to know, if I can use this software for my research. As a developer, I want to know, if anyone is actively maintaining this software
embargoDate	Authors when publishing software via any sort of repository?	Date when the embargo is over	Some software might be restricted for a period of time. The users need to know when this period of restriction has ended.	
OperatingSystem	User/service	Reinstall / reuse	The operating system should be described so the user or service can install or run the tool	An IT user wanting to compare performances of given OS's used in the implementation of some kind of software
SoftwareRequirements	user/other software/dependencies	Install before reuse, Or combine with other tools	To ensure all the pre-requisites for the software are available before re-use To inform about dependencies of the	A user wanting to be sure that the software is not using a dependency which he can not

	Actor - Who?	Action - What?	Reason - Why?	Scenario
<i>metadata</i>	<i>Stakeholder that does the action</i>	<i>Action needed</i>	<i>The goal of the action, why do this actor needs this action</i>	As a [actor] I can [action] so that [reason]
			<i>Resource</i>	use in his own environment (incompatibilities)

Conclusion and next steps (for session A)

Identify open issues, challenges and blockers from exercises

Next steps

How will we proceed to develop this initiation activity to create the FAIR-IMPACT guidelines to metadata curation for research software?

How can we improve the CodeMeta vocabulary and crosswalk tables?

- Update on the v2.1 and v3.0

How can we improve the codemeta-generator tool?

- Open issues
- Submit PRs
- A dedicated activity during the FAIRCORE4EOSC sprint in October 2023

Annexe

Research software definition

*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 specifically for research 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, and hardware) used during the research are identified, described, and made accessible to the extent that is possible.*

FAIR4RS output: Gruenpeter et al. Defining Research Software: a controversial discussion (Version 1). Zenodo.
<https://doi.org/10.5281/zenodo.5504016>

Where is the metadata available ? Platforms/Infrastructures description

<i>Infrastructure/ Platform type</i>	<i>Definition/ Why do we focus on this actor/infrastructure?</i>	<i>Examples</i>
<i>Software development platform</i>	<p>An online service for developers to collaborate on software development activities</p> <p>https://en.wikipedia.org/wiki/Collaborative_development_environment</p> <p>https://en.wikipedia.org/wiki/Version_control</p>	<ul style="list-style-type: none"> ● GitHub ● Bitbucket ● SourceForge ● ...
<i>Scholarly Repositories</i>	<p>“An organisation called to archive and make available research artifacts, e.g. articles, datasets, software.”</p> <p>EOSC Executive Board & EOSC Secretariat. (2020). <i>Scholarly infrastructures for research software. Report from the EOSC Executive Board Working Group (WG) Architecture Task Force (TF) SIRS</i>. European Commission. Directorate General for Research and Innovation. https://data.europa.eu/doi/10.2777/28598</p>	<ul style="list-style-type: none"> ● HAL ● Zenodo ● Dryad ● Your example: URL are welcome
<i>Registries (catalogs)</i>	<p>“<i>Research software registries</i> are typically indexes or catalogs of software metadata, without any code stored in them; while in <i>research software repositories</i>, software is both indexed <i>and</i> stored (Lamprecht et al., 2020).”</p> <p>Garijo, D., Ménager, H., Hwang, L., Trisovic, A., Hucka, M., Morrell, T., & Allen, A. (2022). Nine best practices for research software registries and repositories. <i>PeerJ Computer Science</i>, 8, e1023. https://doi.org/10.7717/peerj-cs.1023</p>	<ul style="list-style-type: none"> ● The DataCite Metadata collection ● Your example: URL are welcome ●
<i>Publishers</i>	<p>“Any organization that prepares submitted research texts, possibly with associated source code and data, to produce a publication and manage its dissemination, promotion, and archival process.”</p> <p>“[...] there is an opportunity for publishers to educate authors on the necessity of sharing software source code and encourage a standard workflow.”</p> <p>EOSC Executive Board & EOSC Secretariat. (2020). <i>Scholarly infrastructures for research software. Report from the EOSC</i></p>	<ul style="list-style-type: none"> ● Dagstuhl ● IPoI ● Your example: URL are welcome ●

Infrastructure/ Platform type	Definition/ Why do we focus on this actor/infrastructure?	Examples
	<p><i>Executive Board Working Group (WG) Architecture Task Force (TF) SIRS. European Commission. Directorate General for Research and Innovation.</i> https://data.europa.eu/doi/10.2777/28598</p>	
<p><i>Aggregators</i></p>	<p>“Aggregators collect, curate, select, present, and aggregate information about research software from various sources to improve findability in diverse communities.”</p> <p>“Any service that collects information about digital content from a variety of sources with the primary goal of increasing its discoverability, and possibly adding value to this information via processes like curation, abstraction, and classification, and linking.”</p> <p>EOSC Executive Board & EOSC Secretariat. (2020). <i>Scholarly infrastructures for research software. Report from the EOSC Executive Board Working Group (WG) Architecture Task Force (TF) SIRS.</i> European Commission. Directorate General for Research and Innovation. https://data.europa.eu/doi/10.2777/28598</p>	<ul style="list-style-type: none"> • OpenAIRE • swMATH.org • Your example: URL are welcome •

Reuse challenges

Dealing with software collapse in a very large ecosystem (much larger than just the academic ecosystem):

Project-specific code	<i>Scripts, notebooks, workflows, ...</i>
Domain-specific tools	<i>GROMACS, MMTK, ... (domain: biomolecular simulation)</i>
Scientific infrastructure	<i>BLAS, HDF5, SciPy, ...</i>
Non-scientific infrastructure	<i>gcc, Python, ...</i>
Operating system	<i>GNU/Linux, ...</i>
Hardware	<i>x86 processor ...</i>

Konrad Hinsen. Dealing With Software Collapse. Computing in Science and Engineering, Institute of Electrical and Electronics Engineers, 2019, 21 (3), pp.104-108. [hal-02117588](https://hal.archives-ouvertes.fr/hal-02117588)

FAIR4RS Principles v1.0

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.

Chue Hong, N. P., 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>