Sustainable and FAIR Software in Research – A New RDMO Catalogue for Software Management Plans

RSE Munich, 15 March 2023

Michael Franke
franke@mpdl.mpg.de
Dr. Yves Vincent Grossmann
grossmann@mpdl.mpg.de

Agenda

1. Research Software
2. Software Management Plans
3. Added Value
4. Realisation with RDMO

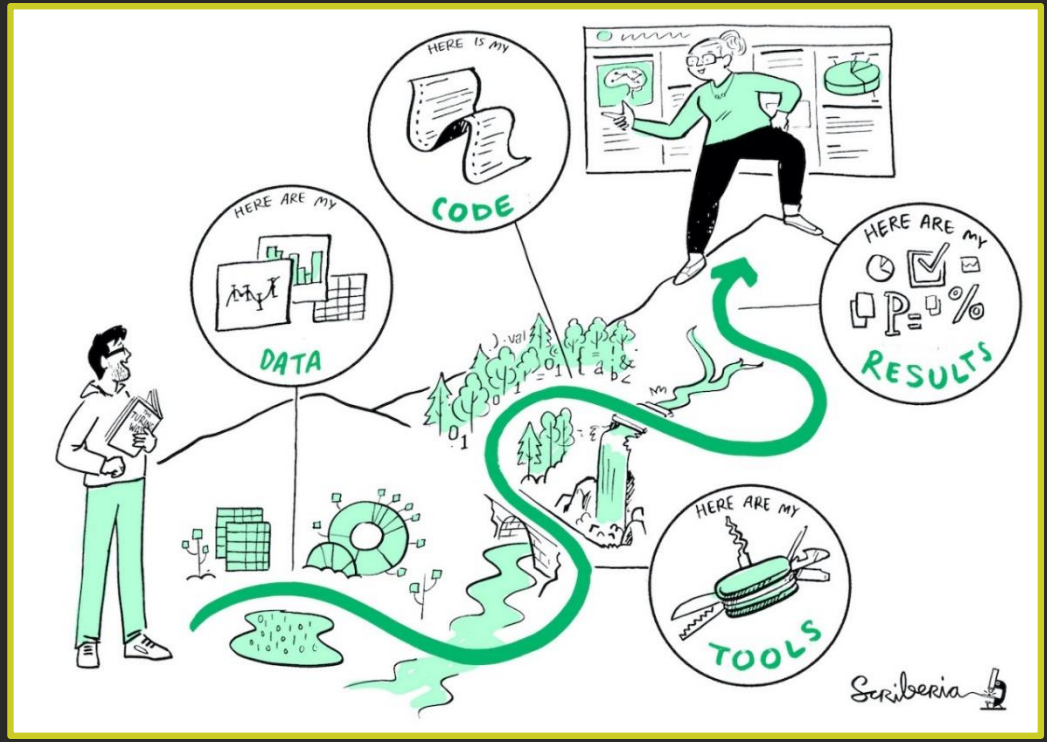
Research Software

Three Pillars of Research Results

Text	Data	Code
• Journal articles • Books • Posters • …	• Raw data • Lab data • Analysed data • … •	• Software • Configuration • Documentation • …

Reproducibility and Accessibility

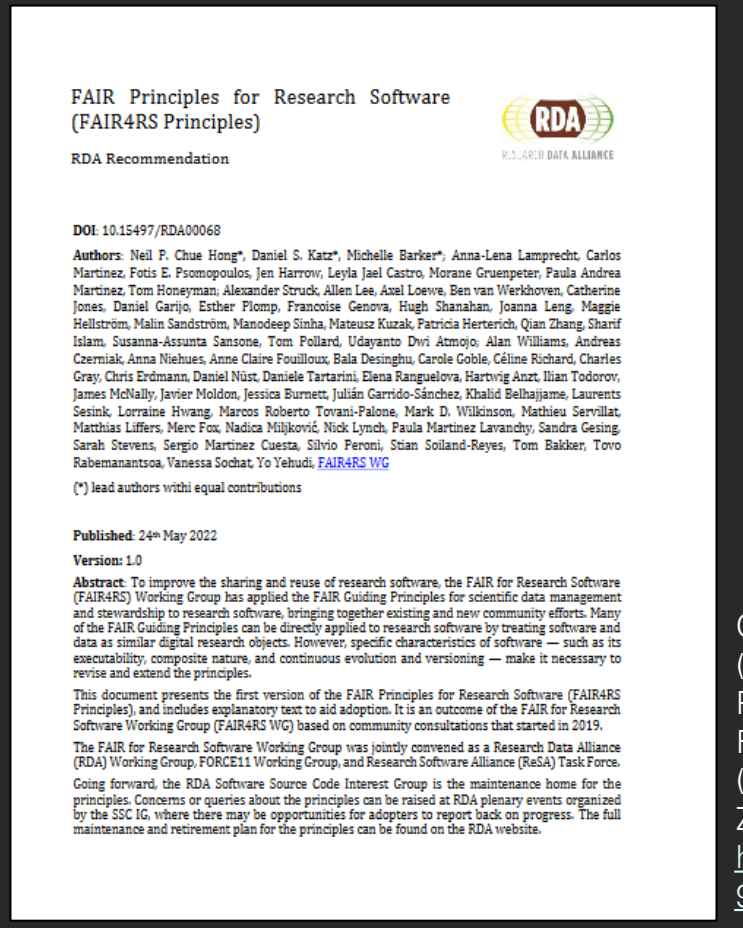
- Software is often needed to reproduce research results
- It should be accessible according to good scientific practice
- Internal policies, funders, journals require or recommend the publication of software
  - *„Software programmed by researchers themselves is made publicly available along with the source code.“*
    Guideline 13: Providing public access to research results in the code of condauct of the DFG



The Turing Way Community, & Scriberia. (2022). Illustrations from The Turing Way: Shared under CC-BY 4.0 for reuse. Zenodo. https://doi.org/10.5281/zenodo.6821117.

FAIR Principles for Research Software (FAIR4RS Principles)

- Adaption of the FAIR data principles to research software
- Summer 2022
- Community-driven



FAIR Principles for Research Software (FAIR4RS Principles)

RDA Recommendation

DOI: 10.15497/RDA00068

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

(*) lead authors withi equal contributions

Published: 24th May 2022

Version: 1.0

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

Going forward, the RDA Software Source Code Interest Group is the maintenance home for the principles. Concerns or queries about the principles can be raised at RDA plenary events organized by the SSC IG, where there may be opportunities for adopters to report back on progress. The full maintenance and retirement plan for the principles can be found on the RDA website.

Chue Hong et al. (2022): FAIR Principles for Research Software (FAIR4RS Principles, Zenodo, CC BY 4.0, https://doi.org/10.15497/RDA00068.

Differences FAIR Data ↔ FAIR4RS

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

- (=) F1 Software is assigned a globally unique and persistent identifier
  - (new) F1.1 Components of the software representing levels of granularity are assigned distinct identifiers
  - (new) 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

Accessible: Software, and its metadata, is retrievable via standardised protocols

- (=) A1 Software is retrievable by its identifier using a standardised 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

Interoperable: 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. (meta)data use vocabularies that follow FAIR principles~~
- (=) I2~~3~~. Software includes qualified references to other objects

Reusable: 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
- (new) R2 Software includes qualified references to other software
- (=) R~~1~~3. Software meets domain-relevant community standards

Folie nach Leyla Jael Castro: FAIR Principles for Research Software, gehalten auf der HMC FAIR Friday Seminar Series am 9. Dezember 2022, Seite 5, https://fdm.uni-koeln.de/sites/FDM-UzK/Data_Champions/C3RDM_Kick_Off_Data_Champions_Keynote_Leyla_Castro_2021-12-09.pdf.

Comparing Research Software and Data (1)

Similarities between software and data in the research process:

- Both are taking more and more place in research
- Both are essential to reproduce research results
- Both offer potentials for re-use
- Both are rarely accepted as independent research efforts

Comparing Research Software and Data (2)

Similarities between software and data in their respective management:

- Both can be archived and be provided with metadata
- Both have to be curated to guarantee usability over longer periods of time
- With both, often management skills are lacking

Comparing Research Software and Data (3)

Differences between research software and data:

- Software curation is much more elaborate
- Versioning plays a significantly bigger role with software
- Software metadata is much more homogenous
- There is a lot of experience with (open source) software licencing

Use of Software Repositories

- Differing publication patterns between software and data
- Our observation: software that is mentioned in a publication is often directly linked to GitHub, GitLab, …
- *„In 2021, one out of five publications in the arXiv corpus included a URI to GitHub“* (S. 1)

arXiv:2208.04895v1 [cs.DL] 9 Aug 2022

The Rise of GitHub in Scholarly Publications*

Emily Escamilla[1][0000−0003−3845−7842], Martin Klein[2][0000−0003−0130−2097], Talya Cooper[3][0000−0003−4241−6330], Vicky Rampin[3][0000−0003−4298−168X], Michele C. Weigle[1][0000−0002−2787−7166], and Michael L. Nelson[1][0000−0003−3749−8116]

[1] Old Dominion University, Norfolk, VA, USA
{evogt001@odu.edu, {mweigle, mln}@cs.odu.edu
[2] Los Alamos National Laboratory, Los Alamos, NM, USA
mklein@lanl.gov
[3] New York University, New York, NY, USA
{tc3602, vm77}@nyu.edu

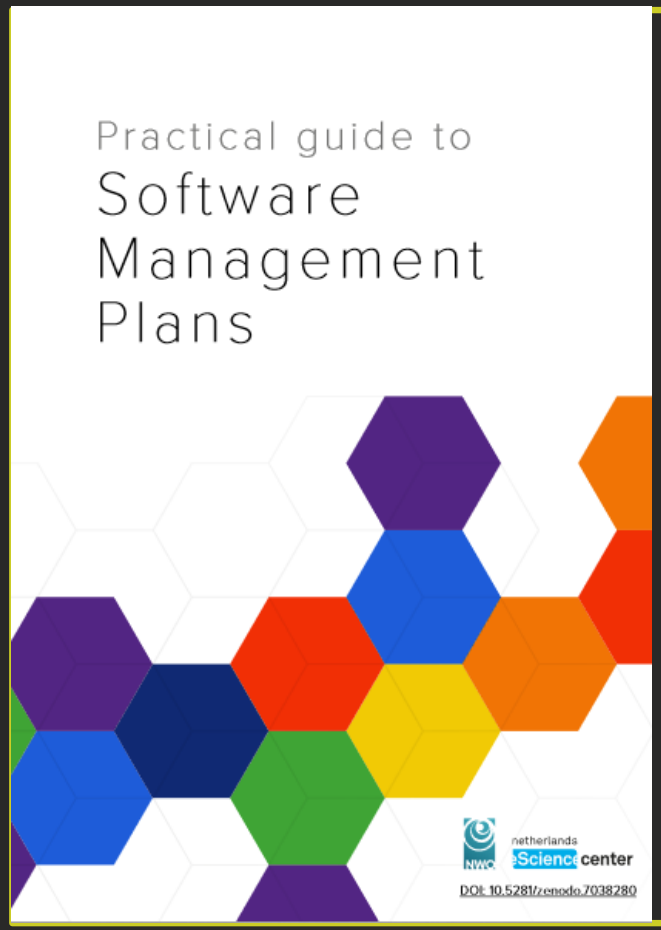
Abstract. The definition of scholarly content has expanded to include the data and source code that contribute to a publication. While major archiving efforts to preserve conventional scholarly content, typically in PDFs (e.g., LOCKSS, CLOCKSS, Portico), are underway, no analogous effort has yet emerged to preserve the data and code referenced in those PDFs, particularly the scholarly code hosted online on Git Hosting Platforms (GHPs). Similarly, the Software Heritage Foundation is working to archive public source code, but there is value in archiving the issue threads, pull requests, and wikis that provide important context to the code while maintaining their original URLs. In current implementations, source code and its ephemera are not preserved, which presents a problem for scholarly projects where reproducibility matters. To understand and quantify the scope of this issue, we analyzed the use of GHP URIs in the arXiv and PMC corpora from January 2007 to December 2021. In total, there were 253,590 URIs to GitHub, SourceForge, Bitbucket, and GitLab repositories across the 2.66 million publications in the corpora. We found that GitHub, GitLab, SourceForge, and Bitbucket were collectively linked to 160 times in 2007 and 76,746 times in 2021. In 2021, one out of five publications in the arXiv corpus included a URI to GitHub. The complexity of GHPs like GitHub is not amenable to conventional Web archiving techniques. Therefore, the growing use of GHPs in scholarly publications points to an urgent and growing need for dedicated efforts to archive their holdings in order to preserve research code and its scholarly ephemera.

Keywords: Web Archiving · GitHub · arXiv · Digital Preservation · Memento · Open Source Software.

* Supported by the Alfred P. Sloan Foundation, https://sloan.org/grant-detail/9628

Software Management Plans

SMPs in Discussion



Martinez-Ortiz et al. (27.10.2022): Practical guide to Software Management Plans, v1.0, https://doi.org/10.5281/zenodo.7248877, CC BY 4.0.

DFG: **Call for Proposals to Increase the Usability of Existing Research Software** durch „Research Software – Quality Assured and Re-usable“, (Information für die Wissenschaft Nr. 85 | 8. November 2022) http://www.dfg.de/en/research_funding/announcements_proposals/2022/info_wissenschaft_22_85.

Definition of a Software Management Plan

„Laut Definition der DINI/nestor AG Forschungsdaten beinhaltet ein Softwaremanagementplan (SMP) allgemeine und technische Informationen zum Softwareprojekt, Angaben zur Qualitätssicherung, zum Release und zur öffentlichen Verfügbarkeit sowie rechtliche und ethische Aspekte, die die Software betreffen.

Der SMP fasst Informationen zusammen, die die Erstellung, Dokumentation, Speicherung, Versionierung, Lizenzierung, Archivierung und/oder Veröffentlichung der in einem Projekt erzeugten oder verwendeten Software hinreichend beschreiben und dokumentieren. Dazugehörige Hardware und notwendige andere Ressourcen, aber auch damit verbundene weitere Software und Softwarebibliotheken, Text- und Datenpublikationen sind ebenfalls zu beschreiben und stellen eine Besonderheit des SMP dar.

Zweck eines SMPs ist zunächst die Nachvollziehbarkeit sowie ggf. die langfristige Nutzbarkeit der Software (zur direkten Anwendung sowie zur Weiterverarbeitung) zu unterstützen und den Support der Nutzer*innen bei Rückfragen zu erleichtern. Der SMP dient folglich auch der Qualitätssicherung (vgl. hierzu FAIR4RS Principles).“

https://forschungsdaten.info/praxis-kompakt/glossar/#c822402

Researchers Writing Research Software

Observations:

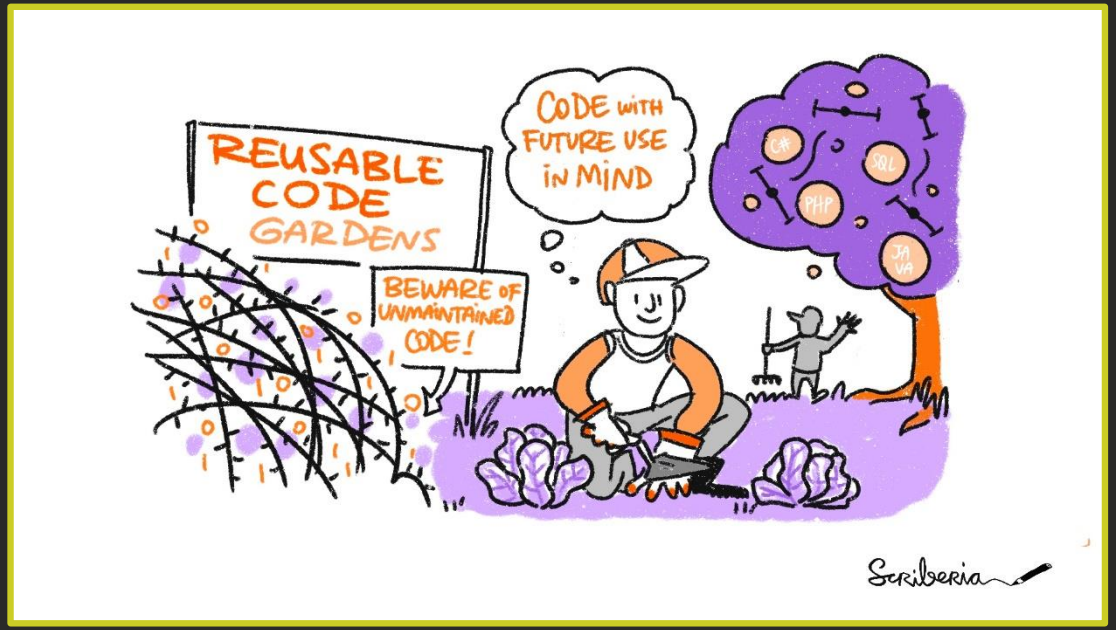
- Usually not thoroughly trained but self-educated developers
- Functionality before documentation before sustainability
- First text publication, then nothing for a long time, and then maybe data publication and software publication
- Software is often handed over from one PHD student to the next
- …

Added Value

Re-Usability of Research Software

Consious handling is increasing the likeliness of re-use:

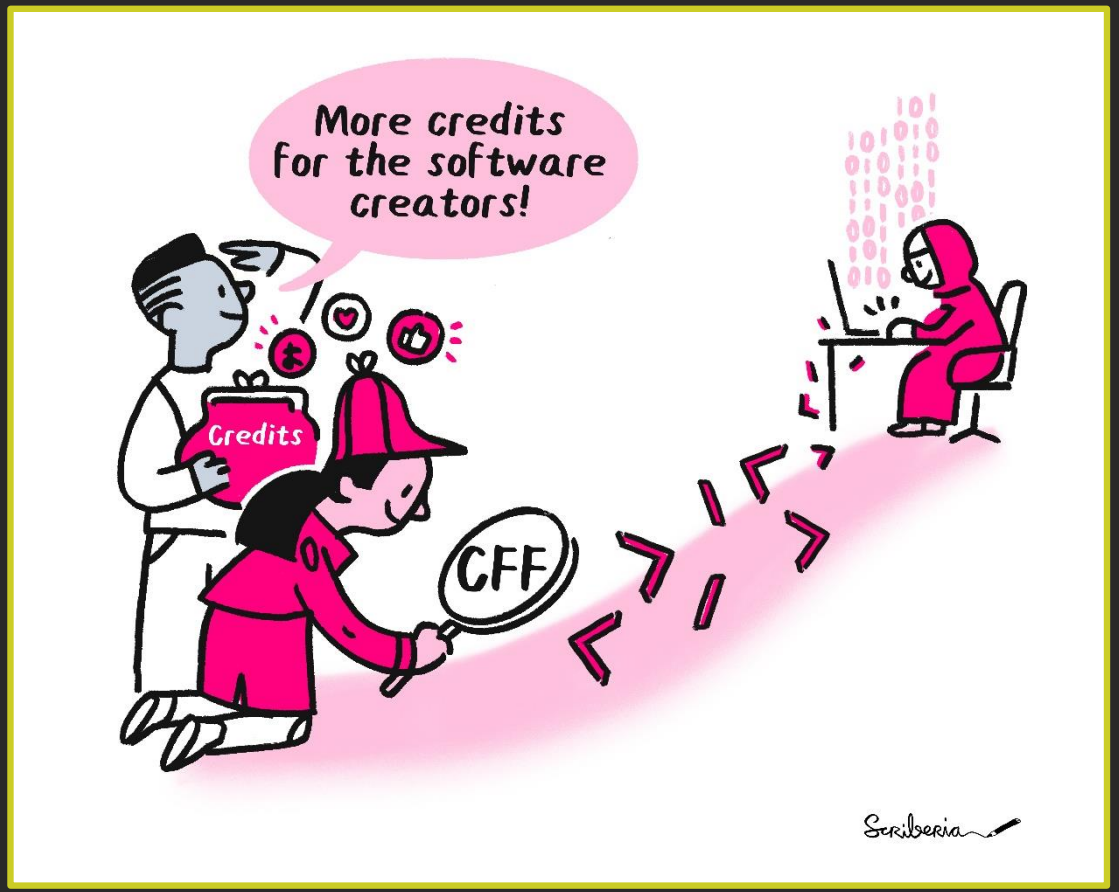
- Increased probability of publication
- Explicit licensing
- Clear code structure and reflected use of third-party libraries
- A targeted approach to archiving
- …



The Turing Way Community, & Scriberia. (2022). Illustrations from The Turing Way: Shared under CC-BY 4.0 for reuse. Zenodo. https://doi.org/10.5281/zenodo.6821117.

Recognition of Research Software

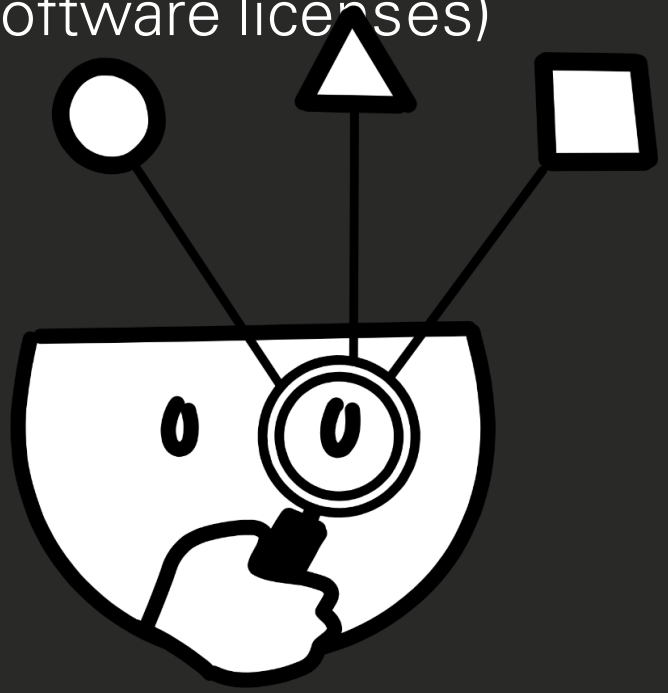
- Disciplinary credits for software publication
- Credits/funding by funders
- Institution-wide credits for development
- Policies needed
  - Normative framework for software publication
  - Endorsement of software publication



The Turing Way Community, & Scriberia. (2022). Illustrations from The Turing Way: Shared under CC-BY 4.0 for reuse. Zenodo. https://doi.org/10.5281/zenodo.6821117.

Added Value for Institutions

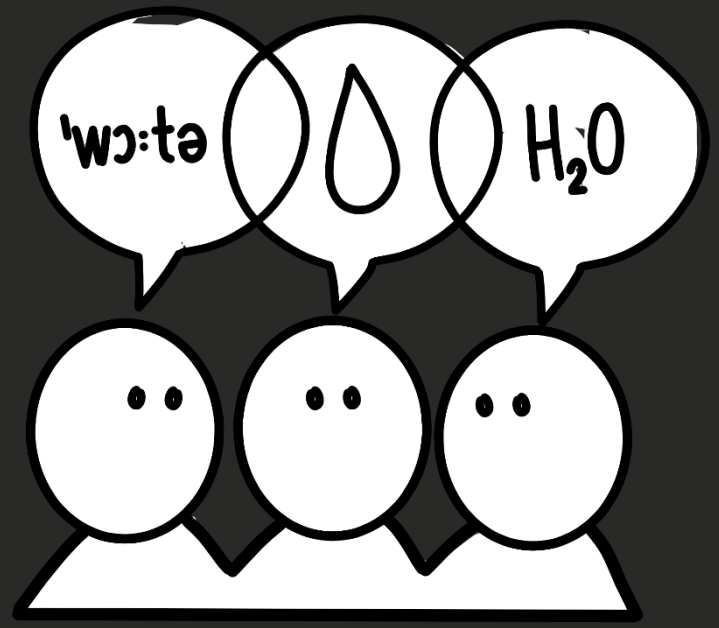
- Support for developers
- Getting an overview on starting/running software projects
- More reasonable planning of demands (e.g. storage, software licenses)
- Better QA
- Easier archiving
- Easier re-use
- …



https://doi.org/10.5281/zenodo.3674561

Why should I write an SMP?

- **For myself!**
- Together with IT/Scientific Computing Unit/… to better design a software project
- For a funding application
- For internal planning
- For sustainability and a possible publication/archiving (good scientific practice)
- Quality assurance
- …



https://doi.org/10.5281/zenodo.3674561

Realisation with RDMO

Project Data

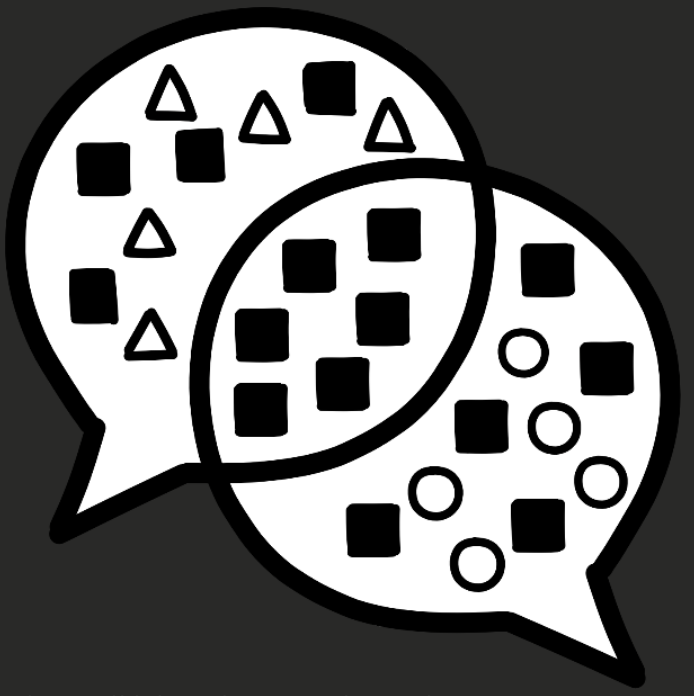
- Max Planck Digital Library, Collections Division
  - Mainly Yves Vincent Grossmann, Michael Franke and Jan Matthiesen
- From July 2022 to December 2022
- RDMO as technical framework
- Outcome: CC0-Push of an SMP catalogue for the RDMO Community

SMP as RDMO Catalogue

- Title: „Software Management Plan for Researchers“
- in German & Englisch
- 49 questions in total
- Available at https://github.com/rdmorganiser/rdmo-catalog with a CC0 waiver
- FAIR4RS Viewer

Audience

- Researchers
- IT staff, scientific core unit, to consult researchers
- Funding acquisition staff
- PIs, research coordination
- …

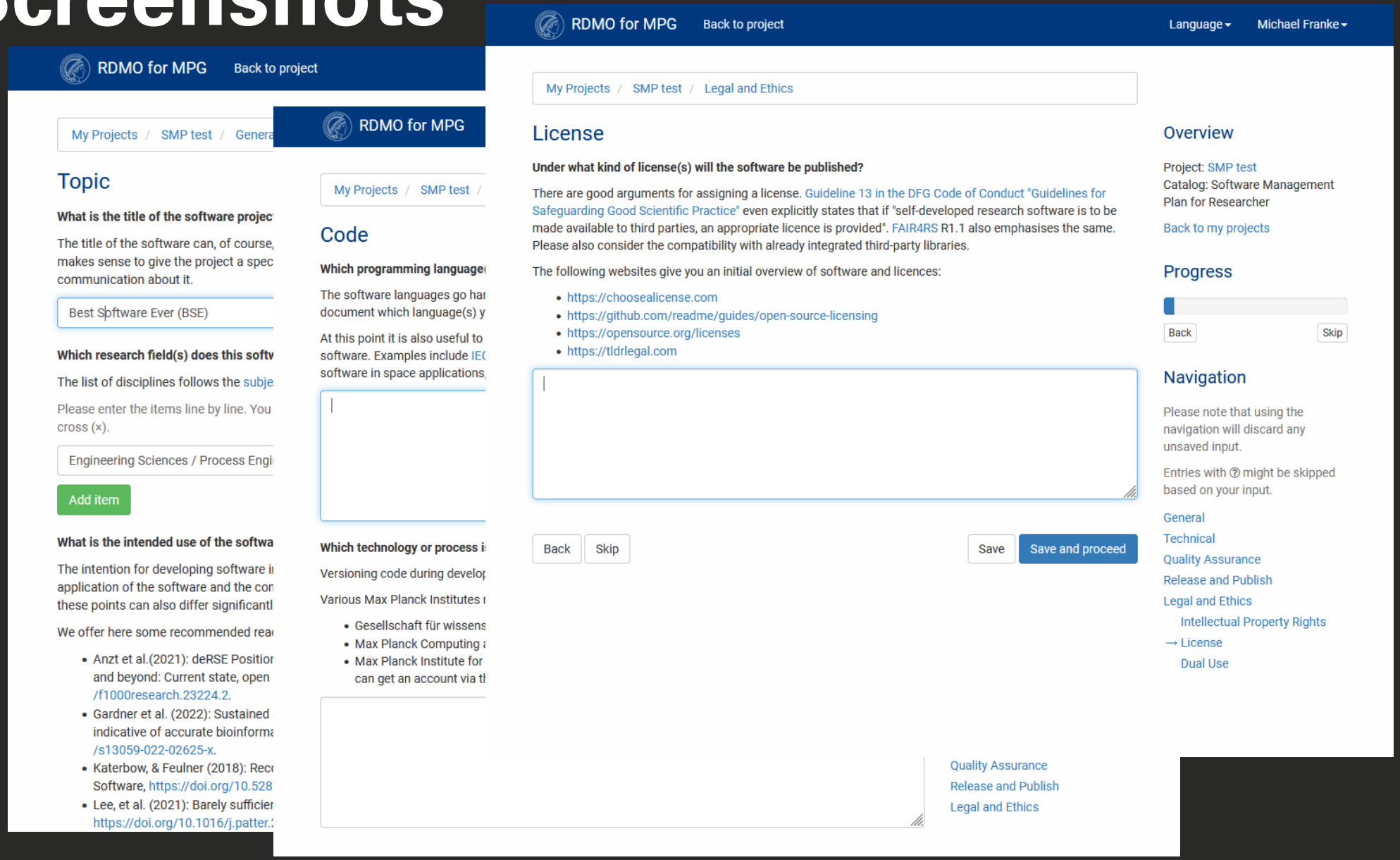


https://doi.org/10.5281/zenodo.2581783

Catalogue Structure

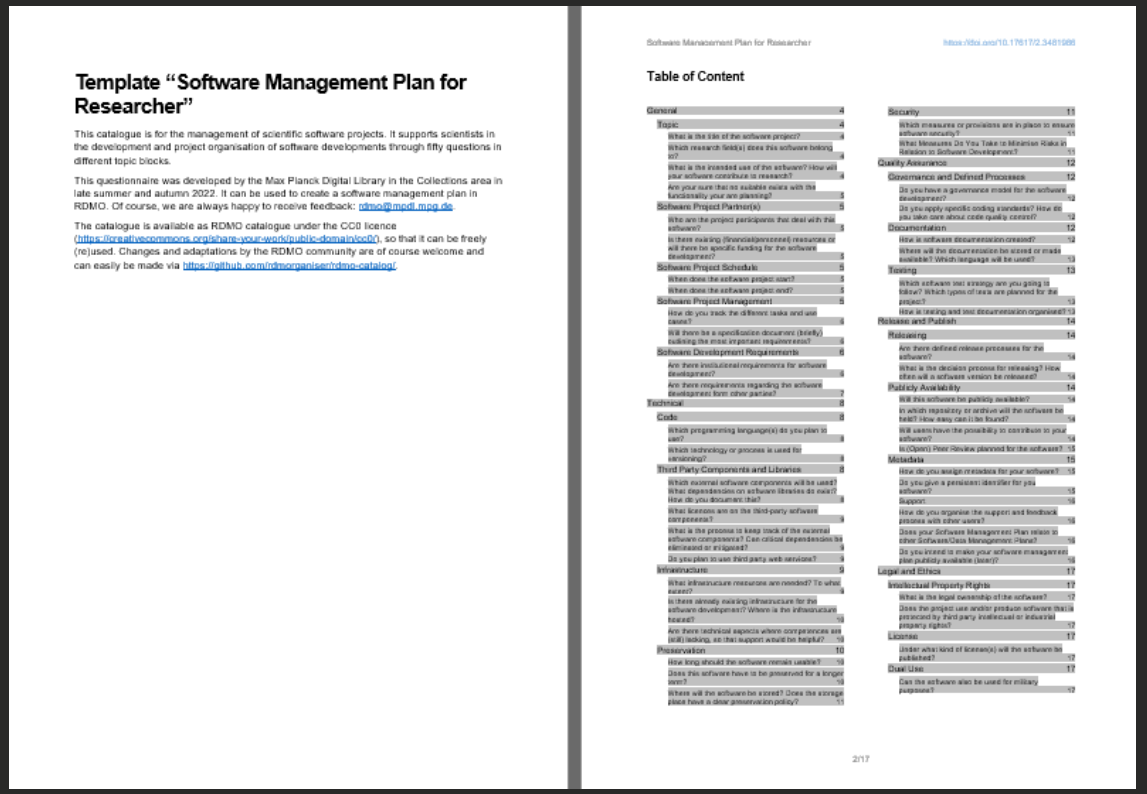
1. General
   - Stakeholders, project management, …
2. Technical
   - Code, Infrastructure, Security/Safety, …
3. Quality Assurance
   - Testing, Documentation, …
4. Release and Publish
   - Releases, metadata, re-use, …
5. Legal and Ethics
   - Copyright, licenses, dual use, …

Screenshots



Questions and Supporting Information as .docx

- Only in English
- CC0 → easily adaptable
- https://doi.org/10.17617/2.3481986



Template "Software Management Plan for Researcher"

This catalogue is for the management of scientific software projects. It supports scientists in the development and project organisation of software developments through fifty questions in different topic blocks.

This questionnaire was developed by the Max Planck Digital Library in the Collections area in late summer and autumn 2022. It can be used to create a software management plan in RDMO. Of course, we are always happy to receive feedback: rdmo@mpdl.mpg.de.

The catalogue is available as RDMO catalogue under the CC0 licence (https://creativecommons.org/share-your-work/public-domain/cc0), so that it can be freely (re)used. Changes and adaptations by the RDMO community are of course welcome and can easily be made via https://github.com/rdmorganiser/rdmo-catalog/.

Software Management Plan for Researcher https://doi.org/10.17617/2.3481986

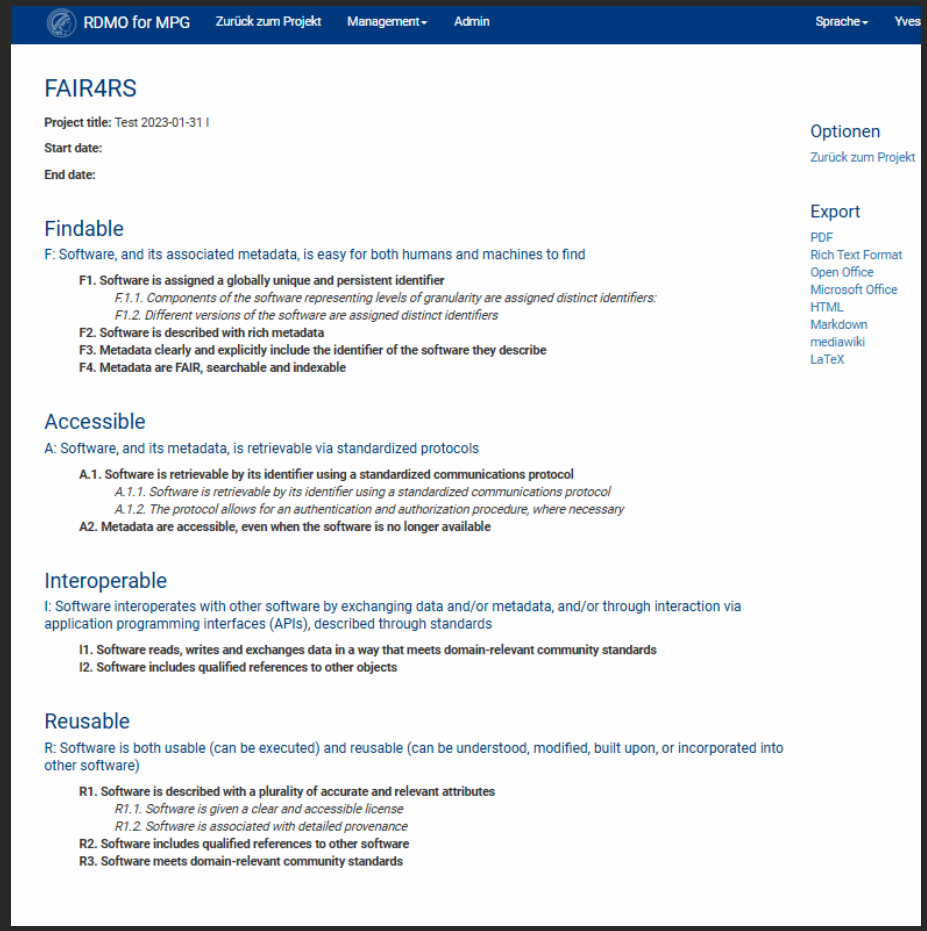
Table of Content

General 4
Topic 4
What is the title of the software project? 4
Which research field(s) does this software belong to? 4
What is the intended use of this software? How will your software contribute to research? 4
Are your sure that no suitable exists with the functionality your are planning? 5
Software Project Partner(s) 5
Who are the project participants that deal with the software? 5
Is there existing (financial/personnel) resources or will there be specific funding for the software development? 5
Software Project Schedule 5
When does the software project start? 5
When does the software project end? 5
Software Project Management 5
How do you track the different tasks and use cases? 6
Will there be a specification document (briefly) outlining the most important requirements? 6
Software Development Requirements 6
Are there institutional requirements for software development? 6
Are there requirements regarding the software development form other parties? 7
Technical 8
Code 8
Which programming language(s) do you plan to use? 8
Which technology or process is used for versioning? 8
Third Party Components and Libraries 8
Which external software components will be used? 8
What dependencies on software libraries do exist? How do you document that? 9
What licenses are on the third-party software components? 9
What is the process to keep track of the external software components? Can critical dependencies be eliminated or mitigated? 9
Do you plan to use third party web services? 9
Infrastructure 9
What infrastructure resources are needed? To what extent? 9
Is there already existing infrastructure for the software development? Where is the infrastructure hosted? 10
Are there technical aspects where competences are (still) lacking, so that support would be helpful? 10
Preservation 10
How long should the software remain usable? 10
Does this software have to be preserved for a longer term? 10
Where will the software be stored? Does the storage place have a clear preservation policy? 11
Security 11
Which measures or provisions are in place to ensure software security? 11
What Measures Do You Take to Minimise Risks in Relation to Software Development? 11
Quality Assurance 12
Governance and Defined Processes 12
Do you have a governance model for the software development? 12
Do you apply specific coding standards? How do you take care about code quality control? 12
Documentation 12
How is software documentation created? 12
Where will the documentation be stored or made available? Which language will be used? 13
Testing 13
Which software test strategy are you going to follow? Which types of tests are planned for the project? 13
How is testing and test documentation organised? 13
Releases and Publish 14
Releasing 14
Are there defined release processes for the software? 14
What is the decision process for releasing? How often will a software version be released? 14
Publicly Availability 14
Will this software be publicly available? 14
In which repository or archive will the software be held? How easy can it be found? 14
Will users have the possibility to contribute to your software? 14
Is (Open) Peer Review planned for the software? 15
Metadata 15
How do you assign metadata for your software? 15
Do you give a persistent identifier for you software? 15
Support 15
How do you organise the support and feedback process with other users? 15
Does your Software Management Plan relate to other Software/Data Management Plans? 16
Do you intend to make your software management plan publicly available (later)? 16
Legal and Ethics 17
Intellectual Property Rights 17
What is the legal ownership of the software? 17
Does the project use and/or produce software that is processed by third party intellectual or industrial property rights? 17
Licenses 17
Under what kind of license(s) will the software be published? 17
Dual Use 17
Can the software also be used for military purposes? 17

2/17

FAIR4RS Viewer

- A view of the answers structured by the FAIR4RS facets
- Quick assessment of a software's FAIRness
- Exportable to many formats
- Developed by Jan Matthiesen (MPDL)
- Pull already requested



Take Home Messages

1. Relevance and recognition of research software will increase
2. Making software management explicit brings added value
3. Software management plans can be one way of making software management explicit

Max Planck digital library

Thanks

to the MPDL Collections Team, the UAG-DMP of the DINI/nestor-AG Forschungsdaten und to the RDMO Community

Max Planck digital library

Thanks for listening!

rdm@mpdl.mpg.de