Five Minutes to Write a Software Management Plan – A Machine-actionable Approach to Simplify the Creation of SMPs

Leyla Jael Castro *[1,2], Lukas Geist [1,2], Esteban Gonzalez [3], Mariaisabel Gonzalez-Ocanto [1], Yves Vincent Grossmann [4], Thomas Pronk [5], Dhwani Solanki [1,2], Carlos Utrilla Guerrero [6], David Wallace [7,8], and Jürgen Windeck [7,8]

[1] ZB MED Information Centre for Life Sciences, Cologne
[2] NFDI4DataScience
[3] Universidad Politécnica de Madrid
[4] Max Planck Digital Library, Munich
[5] Amsterdam University Medical Centers
[6] Delft University of Technology
[7] Technical University of Darmstadt
[8] NFDI4Ing

December 2023

Abstract

The Software Management Plan (SMP) is a relevant tool for handling research software. Despite benefits for research (e.g., low barrier for researchers, promotion of good practices), SMPs are not yet used across the board. A semi-automated approach can solve this problem. In the following document, we discuss a possible workflow for creating machine-actionable SMPs using various tools. This approach was developed during an NFDI4DataScience hackathon at the German National Library of Medicine (ZB Med) - Information Centre of Life Sciences on maSMPs at the end of 2023.

*Corresponding author: ljgarcia@zbmed.de.

Contents

1 Introduction

Research software is attracting more and more attention as it is used to create, collect and transform other research artefacts (most notably data). In addition to scholarly and data publications, code is increasingly recognised as an essential scientific product. By extension, ensuring that research software is robust, maintainable, and findable, accessible, interoperable and reusable (FAIR)[1], is also becoming increasingly important. Similar to Data Management Plans for data, SMPs offer a systematic way to ensure the above for software.[2]

Despite the increasing recognition of the role of research software in research, e.g., in 2021 one fifth of arxiv publications referred to software in Git repositories[3], and the emergence of SMPs some years ago[4], the adoption of SMPs is still lacking. Perhaps in part due to the novelty of research software as a first-class research product, it has been difficult to convince scientists of the value of managing their research software. Hence, it is a worth-wile pursuit to ensure that SMPs are as relevant and valuable as they can be. To make SMPs more relevant, they could be tailored to the specific needs of particular types of research software.[5] SMPs could also complement efforts compiling software produced by an institution.[6]. To make SMPs more valuable, they can be expressed in universal standards, such as the maSMP metadata schema [7].

In addition, adoption of SMPs can be facilitated by saving the researcher work in producing them. This is already achieved in part by improving relevancy, thereby having researchers only produce the parts that apply to their use cases. Tools such as the Research Data Management Organiser (RDMO)[8] can aid such tailoring, by offering wizard and drill-down methods. Another approach is by automating the SMP production process as much as possible, by reusing information that is already available. Tools such as the SOMEF can do so by extracting relevant software metadata from git repositories.

As yet, maSMP, RDMO, SOMEF, have been operating in isolation of each other. However, their qualities are clearly synergistic, so linking them together could make for an SMP production workflow that is more relevant, more valuable, and more efficient. In the rest of this manuscript, we report on the combination of maSMP, RDMO, SOMEF, with a focus on the achievments carried off during the NFDI4DS[9] maSMP hackathon at ZB MED[10]. We start with a short introduction to the maSMP. We continue with the integration of SMPs in RDMO, including the use of SOMEF and citation files, and improvements to the SMP template created by the Max Planck Digital Library (MPDL)[11]. We conclude with some final thougths on future work and how to move forward.

[1] Barker et al. 2022.

[2] See for example Alves et al. 2021 and Martinez-Ortiz et al. 2023.

[3] Escamilla et al. 2022.

[4] The Software Sustainability Institute 2018

[5] For example the Dutch eScience SMPs in Martinez-Ortiz et al. 2023 and German Aerospace Center (DLR) application classes Schlauch, Meinel, and Haupt 2018, pp. 7–8.

[6] https://github.com/jmelot/SoftwareImpactHackathon2023_InstitutionalOSS.

[7] For example Castro et al. 2023.

[8] Klar et al. 2023.

[9] https://www.nfdi4datascience.de.

[10] https://www.zbmed.de.

[11] https://www.mpdl.mpg.de.

2 maSMP metadata schema

The idea behind maSMP builds on top of the machine-actionable Data Management Plans[12] proposed by the Research Data Alliance DMP Common Standards working group[13]. The added value of a standard for maSMPs lies firstly in the interoperability. The data flow from and to SMPs is thereby automatically enabled or increased. At the same time, this (supplementary) vocabulary makes it possible to better describe an SMP and the associated software. This is what makes it possible to use tools such as SOMEF on a large scale in the first place. Adding a machine-actionable layer on top of SMP should make it easier to, for instance, compare SMPs created by different parties, and facilitate the combination of SMPs with other management plans and corresponding outputs.

An initial SMP metadata schema [14] was developed in the context of Research Data Alliance and European Open Science Cloud Future in the year 2022. It focused on the software handled by the management plan, reused schema.org[15], and took the ELIXIR SMP[16] as a template for the needed metadata properties. This schema was further developed thanks to the crosswalks created during a workshop at ZB Med in summer 2023[17] and within the scope of the German National Data Research Infrastructure (NFDI)[18].

2.1 Crosswalks for SMPs

Right before the hackathon, the maSMP metadata schema was migrated[19] to the Data Discovery Engine[20] to better align it to the internal structure of schema.org. During the NFDI4DS hackathon at ZB Med, this schema vr2. was used for crosswalks to the SMP by MPDL[21] and ELIXIR [22], and SOMEF. Furthermore, the schema was enriched with a representation of the actual plan (i.e., the questionnaire corresponding to the plan) which was mapped to `OutputManagementPlan` type in the DataCite Metadata schema (vr.4.4)[23]. Elements from the mapped resources that did not matched were further analysed and some of them were integrated to the schema. The summary of new properties is shown in Table 1, two properties are borrowed from CodeMeta (with an extended range for `codemeta:buildInstructions`) and six are proposed for the maSMP metadata schema; namespace for types and properties from schema.org is omitted. The crosswalks are publicly available as Comma Separated Value files[24]. By the time of

[12] Miksa, Walk, and Neish 2020.

[13] `https://www.rd-alliance.org/groups/dmp-common-standards-wg`

[14] Giraldo, Cardoso, et al. 2023 and Giraldo, Lukas, et al. 2023.

[15] `http://schema.org`, see also Guha, Brickley, and Macbeth 2016.

[16] Alves et al. 2021.

[17] Giraldo, Cardoso, et al. 2023.

[18] Giraldo, Lukas, et al. 2023

[19] `https://discovery.biothings.io/ns/maSMP`

[20] `https://discovery.biothings.io`.

[21] Grossmann 2022.

[22] Alves et al. 2021.

[23] DataCite Metadata Working Group 2021.

[24] Castro et al. 2023

Table 1: New properties added to the maSMP metadata schema.

Property	Description	Range
	SoftwareSourceCode and SoftwareApplication	
codemeta:buildInstructions	Link to installation instructions/documentation	Text or URL
codemeta:issueTracker	Link to software bug reporting or issue tracking system	URL
maSMP:changelog	Link to CHANGELOG file	URL
maSMP:intendedUse	Purpose and intended use of this software	Text or URL
maSMP:deployInstructions	Text or link to deployment instructions/documentation	Text or URL
maSMP:installInstructions	Text or link to installation instructions/documentation	Text or URL
maSMP:testInstructions	Text or link to testing instructions/documentation	Text or URL
	maSMP:SoftwareTestAction	
maSMP:softwareTested	Link to the software tested by this action	SofwareSourceCode or SoftwareApplication

writing, the maSMP metadata schema was at vr2.1.0 including only types (with profiles, i.e., usage recommendations on properties from parent types and own properties, in preparation).

2.2 Mapping between SOMEF and maSMP

After defining a mapping [25] between SOMEF and the model proposed by maSMP, we decide to develop a script to generate the SMP based on the results provided by SOMEF. The code is available here: `https://github.com/oeg-upm/somef2smp`. To test it, we have executed SOMEF on the repositories of the MPDL. All outputs in JSON format have been uploaded to the repository. The script can generate different SMP plans depending on the template used. The mapping between SOMEF outputs fields and the attributes in the JSON-LD are configured by a json file. Results have been uploaded to the repository.

2.3 Use of SOMEF and maSMP in ZB Med

The Semantic Technologies team at ZB Med uses GitHub pages to provide insights of its research projects together with metatada using types from schema.org and profiles from Bioschemas[26]. Outcomes produced within the scope of a research project are packed using RO-Crates[27]. At this point the GitHub pages are work in progress and most of the metadata has been manually created. During the hackathon, we looked into ways to use SOMEF to extract metadata corresponding to GitHub repositories created within the team. The process followed for the extraction is shown in Figure 1. The integration of maSMP together with its corresponding metadata is next on the agenda.

3 Work on SMPs in RDMO

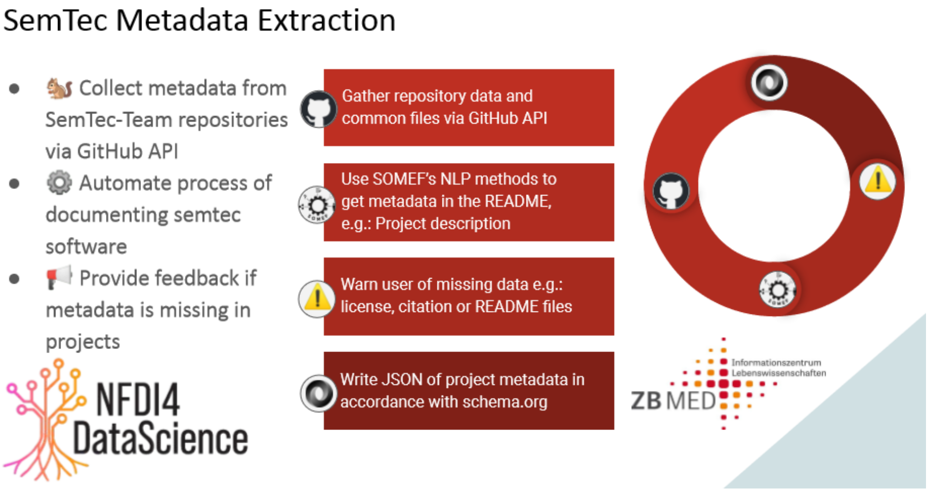
The RDMO is a tool for data management planning used widely across Germany. It is a community-based Open Source software.[28] In November 2022 a team from the MPDL created a

[25] `https://docs.google.com/spreadsheets/d/1BMsc3SKclf5PHVLc-QILY_YWuF_gqcE97qkgOWZpRrE/`.

[26] Gray, Goble, and Jimenez 2017.

[27] Soiland-Reyes et al. 2022.

[28] For all the code and more material see `https://github.com/rdmorganiser`.



SemTec Metadata Extraction

- Collect metadata from SemTec-Team repositories via GitHub API
- Automate process of documenting semtec software
- Provide feedback if metadata is missing in projects

Gather repository data and common files via GitHub API

Use SOMEF's NLP methods to get metadata in the README, e.g.: Project description

Warn user of missing data e.g.: license, citation or README files

Write JSON of project metadata in accordance with schema.org

Figure 1: Metadata extraction at ZB Med for GitHub repositories using SOMEF

SMP template for RDMO.[29] The template consists of 50 questions covering diverse aspects of the research software development process. Users can organise their SMPs in RDMO in so-called projects.

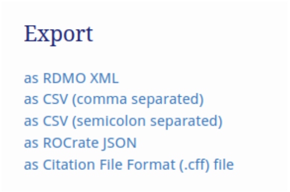
3.1 SOMEF-Plugin for RDMO

Creating a comprehensive SMP is a great challenge for researchers having limited time resources. For this reason, all of the already existing resources and information on their software project stored in different locations and documents should be utilised. These information could be extracted, e.g. from existing Github repositories. Since SOMEF allows to extract such information, we have taken steps to integrate SOMEF into RDMO.

As a first step, we created a mapping between the SOMEF json output file and the internal vocabulary used by the SMP template in RDMO. The RDMO vocabulary is a flexible list of so-called attributes that can be defined for specific use cases, but also consists of standardised URIs maintained by the RDMO community. We were able to map 17 of the SOMEF metadata fields to the SMP template.

As a second step, we created an RDMO import plugin that reads a SOMEF json file and stores the information in the respective SMP project. Of course, this was still a manual process, since users have to run SOMEF first and collect the json file in order to use it as import in RDMO. So as the last step, we implemented a new form in RDMO allowing to add a GitHub link directly to the project. SOMEF is then automatically called and the resulting json file is directly imported into the project. This reduces the time and effort needed and provides a more user-friendly and

[29] https://github.com/rdmorganiser/rdmo-catalog/pull/165. For more see i.e. Grossmann and Franke 2023.



Export

as RDMO XML
as CSV (comma separated)
as CSV (semicolon separated)
as ROCrate JSON
as Citation File Format (.cff) file

Figure 2: Screenshot from the .cff-plugin in the RDMO UI

fast solution. Researchers then start their SMP creation process with a pre-filled plan. The plugin ist available at `https://github.com/rdmorganiser/rdmo-plugins-somef`

3.2 .cff-Plugin for RDMO

The Citation File Format (.cff) format is a human- and machine-readable format for assigning metadata to software in yaml. It was developed in 2021 by a group around Stephan Druskat. It supports scientists in the simple allocation of metadata for their research software. It therefore offers a simple way to provide a lot of necessary information about the code of a research project.[30] The .cff format is now supported by many popular programmes and services such as Zotero, Zenodo and GitHub.com.

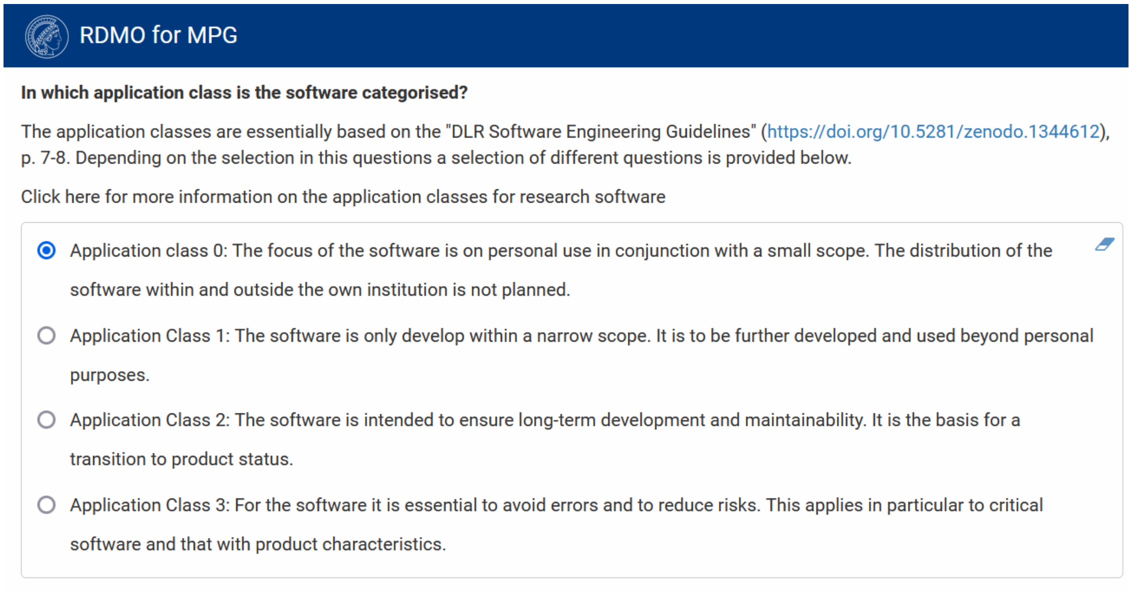
The .cff format makes it easy for scientists to assign metadata to their research software. This information already exists in the SMP if it was filled out in RDMO. We have therefore developed an export plugin that uses the information elements in the SMP catalogue to create the corresponding .cff file, see Figure 2. The plugin addresses the required attributes in RDMO and writes them to the corresponding .cff fields. As a result, scientists can download a CITATION.cff file and add it to their software. The plugin is available at
`https://github.com/rdmorganiser/rdmo-plugins-cff`

3.3 New Iterations for the SMP-Template in RDMO

Since the publication of the SMP template created by MPDL by the end of 2022 there was some feedback that could be incorporated. The aspect of software scalability remained open for a long time. A script does not require the same level of management as an entire infrastructure. To better represent this, the SMP in RDMO was enabled for this in a further iteration.

The selection and categorisation of research software was based on the DLR application classes.[31], see Figure 3. Class 0 refers primarily to scripts to process data for a publication. Application class 1 is primarily aimed at final theses and code demonstrators. Class 2 means software that offers a wide range of functions and expands the system in the long term. This can be a dissertation or a third-party funded project, for example. Finally, class 3 is intended for large infrastructure projects. The mapping from SMP questions and applications classes is documented in the Table 2.

[30] Druskat et al. 2021 and Druskat 2023. See also `https://citation-file-format.github.io`.
[31] Schlauch, Meinel, and Haupt 2018, pp. 7–8.



RDMO for MPG

In which application class is the software categorised?

The application classes are essentially based on the "DLR Software Engineering Guidelines" (https://doi.org/10.5281/zenodo.1344612), p. 7-8. Depending on the selection in this questions a selection of different questions is provided below.

Click here for more information on the application classes for research software

● Application class 0: The focus of the software is on personal use in conjunction with a small scope. The distribution of the software within and outside the own institution is not planned.

○ Application Class 1: The software is only develop within a narrow scope. It is to be further developed and used beyond personal purposes.

○ Application Class 2: The software is intended to ensure long-term development and maintainability. It is the basis for a transition to product status.

○ Application Class 3: For the software it is essential to avoid errors and to reduce risks. This applies in particular to critical software and that with product characteristics.

Figure 3: Screenshot from the RDMO Question on Application Classes for the Research Software

To achieve this for the SMP in RDMO we introduced one attribute, three conditions and the corresponding options. All the work was bundled together and transferred to the RDMO community in a pull request.[32] Thanks to these adjustments to the SMP catalogue, it is now possible to adapt the plan depending on the scope. Basically, there are always nine questions to answer. In category 1, there are 30 questions to answer. In category 2 there are 47 questions. And for category 3, there are now 50 questions to complete.

[32] https://github.com/rdmorganiser/rdmo-catalog/pull/244.

4 Conclusion

SMPs are a useful instrument for fostering research software as a first-class research product. During the hackathon we were able to map different maSMP formats, via which we could connect several complementary approaches and tools for SMPs. The mappings will ensure interoperability for the different SMP templates and structures. The availability of a more comprehensive toolchain will make it faster and easier to produce SMPs, as well as increase the value SMPs add to the research cycle.

1. We have created the basis for the metadata flow by expanding the maSMP metadata schema. An automatic metadata flow is now possible.

2. SOMEF makes it easier for researchers to automatically extract software related metadata from GitHub repositories so it can be used to feed into the corresponding maSMP metadata and RDMO.

3. We adjusted the RDMO SMP template to take into account the specific needs of researchers writing software of different levels that can be described as software application classes. The SMP is now scaling on this classes to better align with the needs of different researchers.

4. The new created .cff file export for RDMO allows to reuse the metadata created in the RDMO in other contexts. Scientists can thus directly reuse their information from the SMP in RDMO and use it for citation in GitHub, Zenodo, etc., for example. There is no intermediate step to create the .cff file.

With our approach we manage to significantly reduce the time resources needed to create a comprehensive software management plan and reusing it.

These following next steps look promising to address:

1. Other options to extract metadata from software repositories can be combined within the RDMO plugin in the future. This can be for example from the HERMES project[33] or the Linking Research Software to Research Organizations[34]. These tools could also be nested one behind the other to achieve even better data quality.

2. We have focused on GitHub.com, but there are of course other software repositories. It is therefore important to find out what options SOMEF offers for other repositories and how quickly the specifications can be adapted for this.

3. The maSMP ontology is not yet directly implemented in SOMEF. The next step could therefore be to implement it. This would allow the software-related information from SOMEF to be used much more generically for maSMP applications and beyond.

4. The maSMPs could also be used in practical applications for populating knowledge graphs. In turn, these graphs could be employed to improve research. For instance, they can be used in to enrich the information offered in research software repositories, such as via the Software Catalog Creator[35].

[33] `https://github.com/hermes-hmc`.

[34] `https://github.com/jmelot/SoftwareImpactHackathon2023_InstitutionalOSS`.

[35] Arroyo Márquez 2023.

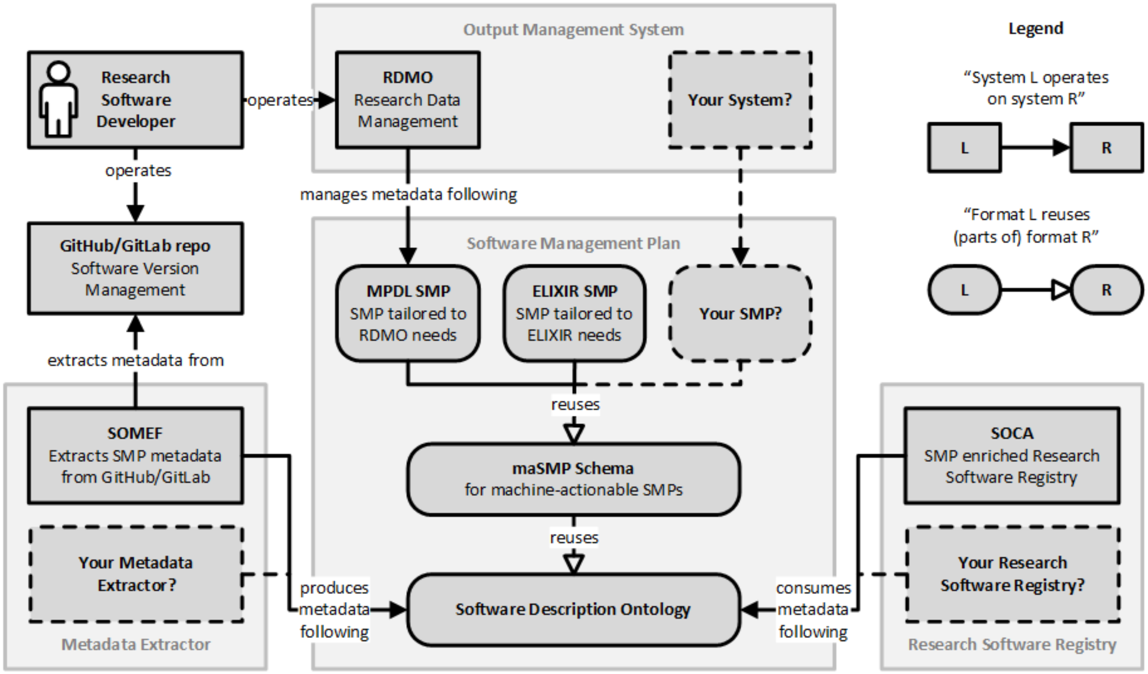

Output Management System
Research Software Developer
operates
RDMO Research Data Management
Your System?
operates
manages metadata following
GitHub/GitLab repo Software Version Management
Software Management Plan
MPDL SMP SMP tailored to RDMO needs
ELIXIR SMP SMP tailored to ELIXIR needs
Your SMP?
extracts metadata from
reuses
SOMEF Extracts SMP metadata from GitHub/GitLab
maSMP Schema for machine-actionable SMPs
SOCA SMP enriched Research Software Registry
reuses
Your Metadata Extractor?
produces metadata following
Software Description Ontology
consumes metadata following
Your Research Software Registry?
Metadata Extractor
Research Software Registry
Legend
"System L operates on system R"
L
R
"Format L reuses (parts of) format R"
L
R

Figure 4: A rough sketch of a possible architecture for integrating software version management systems, metadata extractors, output management systems, and research software registries, using a family of SMP formats

Looking further ahead, we imagine an increasingly integrated SMP toolkit, offering a variety of ways to produce, manage, and use them. We imagine these tools to operate on a range of open standards that are compatible with eachother, so that data structures can both be adapted to individual use cases as well be interoperable. Importantly, the use of open shared standards makes it relatively easily to join the toolchain ecosystem. See Figure 4 for an illustration.

Finally, we briefly reflect on the hackathon format. It brought together a small team with complementary tools, standards, and expertise, who until then operated in isolation of eachother. We experienced the hackathon as a useful way to bootstrap a collaboration, as it focused our initial efforts on delivering a concrete product in a short timeframe. Hence, a four-day hackathon can be a useful step towards achieving a five-minute SMP.

5 Supplements

5.1 Tables

Table 2: Application classes and questions in RDMO

RDMO Question	Application Class
What is the title of the software project?	0
Which research field(s) does this software belong to?	0
What is the intended use of the software? How will your software contribute to research?	0
What function does the new software have that previous software does not cover? What related software exists and why is it not suitable?	0
Who are the project participants that deal with this software?	2
Is there existing (financial/personnel) resources or will there be specific funding for the software development?	1
When does the software project start?	1
When does the software project end?	1
Which software development process is defined? How will process roles be assigned?	2
How do you track the different tasks and use cases?	1
Will there be a specification document (briefly) outlining the most important requirements?	1
Are there institutional requirements for software development?	1
Are there requirements regarding the software development form other parties?	1
Which programming language(s) do you plan to use?	0
Which technology or process is used for versioning?	1
Which external software components will be used? What dependencies on software libraries do exist? How do you document this?	1
What licences are on the third-party software components?	2
What is the process to keep track of the external software components? Can critical dependencies be eliminated or mitigated?	2
Do you plan to use third party web services?	1
Does the software refer to other software projects or objects?	0
What infrastructure resources are needed? To what extent?	1
Is there already existing infrastructure for the software development? Where is the infrastructure hosted?	1
Are there technical aspects where competences are (still) lacking, so that support would be helpful?	1
How long should the software remain usable? What steps will be taken to ensure that the software can continue to be used after the project?	1
Does this software have to be preserved for a longer term?	1
Which measures or provisions are in place to ensure software security?	2
What measures do you take to minimise risks in relation to software development?	3
Do you have a governance model for the software development?	2

Table 2: Application classes and questions in RDMO

RDMO Question	**Application Class**
Do you apply specific coding standards? How do you take care about code quality control?	3
How is software documentation created?	1
Where will the documentation be stored or made available? Which language will be used?	2
Which software test strategy are you going to follow? Which types of tests are planned for the project?	2
Are there defined release processes for the software?	2
What is the decision process for releasing? How often will a software version be released?	2
Where will the software be stored? Does the storage place have a clear preservation policy?	1
Will this software be publicly available?	1
In which repository or archive will the software be held? How easy can it be found? What strategy is used to keep the software usable?	1
Will users have the possibility to contribute to your software?	2
Is (Open) Peer Review planned for the software?	2
How do you assign metadata for your software?	0
Do you give a persistent identifier for you software?	1
Do you plan to give support or help to re-users of your software?	2
How do you organise the support and feedback process with other users?	2
Does your Software Management Plan relate to other Software/Data Management Plans?	1
Do you intend to make your software management plan publicly available (later)?	2
What is the legal ownership of the software?	1
Does the project use and/or produce software that is protected by third party intellectual or industrial property rights?	2
Under what kind of license(s) will the software be published?	0
Can the software also be used for military purposes?	3

6 Acknowledgement

All activities reported here were carried out during the NFDI4DS maSMP hackathon from 28th November to 1st Dezember 2023 at ZB Med. This hackathon was organised by the Semantic Technolgies team and sponsored by NFDI4DataScience. NFDI4DataScience is a consortium funded by the German Research Foundation (DFG), project number 460234259.

7 Abbreviations

DLR German Aerospace Center

maSMP machine-actionable SMP

MPDL Max Planck Digital Library

RDMO Research Data Management Organiser

SOMEF Software Metadata Extraction Framework

SMP Software Management Plan

ZB Med German National Library of Medicine

References

Alves, Renato et al. (2021). *ELIXIR Software Management Plan for Life Sciences.* DOI: 10.37044/osf.io/k8znb.

Arroyo Márquez, Miguel (2023). “Creation of a framework for the monitoring and adoption of good Open Science practices in software repositories”. In.

Barker, Michelle et al. (2022). “Introducing the FAIR Principles for research software”. In: *Scientific Data* 9.1, p. 622. DOI: 10.1038/s41597-022-01710-x.

Castro, Leyla Jael et al. (2023). “A metadata schema for machine-actionable Software Management Plans”. In: DOI: 10.4126/FRL01-006444988.

DataCite Metadata Working Group (2021). *DataCite Metadata Schema Documentation for the Publication and Citation of Research Data and Other Research Outputs Version 4.4.* DOI: 10.14454/3w3z-sa82.

Druskat, Stephan (2023). *The Citation File Format: Providing citation metadata for research software.* DOI: 10.5281/zenodo.7655140.

Druskat, Stephan et al. (2021). “Citation File Format”. In: DOI: 10.5281/zenodo.1003149.

Escamilla, Emily et al. (2022). “The Rise of GitHub in Scholarly Publications”. In: DOI: 10.48550/arxiv.2208.04895.

Giraldo, Olga, João Cardoso, et al. (2023). “Workshop machine-actionable Software Management Plans”. In: DOI: 10.5281/zenodo.8087357.

Giraldo, Olga, Geist Lukas, et al. (2023). “machine-actionable Software Management Plan Ontology (maSMP Ontology)”. In: DOI: 10.5281/zenodo.8089518.

Gray, Alasdair J G, Carole Goble, and Rafael C Jimenez (2017). “From Potato Salad to Protein Annotation”. In: *ISWC Posters and Demo session.* Vienna, Austria, p. 4. URL: http://ceur-ws.org/Vol-1963/paper579.pdf.

Grossmann, Yves Vincent (2022). *Pull Request #165: Add Catalogue for Software Management Plan.* URL: https://github.com/rdmorganiser/rdmo-catalog/pull/165 (visited on 12/01/2023).

Grossmann, Yves Vincent and Michael Franke (2023). “Sustainable Research Software Through Software Management Plans”. In: DOI: 10.17617/2.3525108.

Guha, R. V., Dan Brickley, and Steve Macbeth (2016). “Schema.org: evolution of structured data on the web”. In: *Communications of the ACM* 59.2, pp. 44–51. DOI: 10.1145/2844544.

Klar, Jochen et al. (2023). *Research Data Management Organiser (RDMO).* DOI: 10.5281/zenodo.596581. URL: https://github.com/rdmorganiser/rdmo.

Martinez-Ortiz, Carlos et al. (2023). *Practical guide to Software Management Plans.* DOI: 10.5281/zenodo.7248877.

Miksa, Tomasz, Paul Walk, and Peter Neish (2020). *RDA DMP Common Standard for Machine-actionable Data Management Plans.* DOI: 10.15497/rda00039.

Schlauch, Tobias, Michael Meinel, and Carina Haupt (2018). *DLR Software Engineering Guidelines.* DOI: 10.5281/zenodo.1344612.

Soiland-Reyes, Stian et al. (2022). “Packaging research artefacts with RO-Crate”. In: *Data Science*, pp. 1–42. DOI: 10.3233/DS-210053.

The Software Sustainability Institute (2018). *Checklist for a Software Management Plan.* DOI: 10.5281/zenodo.2159713.