Nine best practices for research software registries and repositories

Daniel Garijo [Corresp., 1] , **Hervé Ménager** [2] , **Lorraine Hwang** [3] , **Ana Trisovic** [4] , **Michael Hucka** [5] , **Thomas Morrell** [5] , **Alice Allen** [6]

[1] Universidad Politécnica de Madrid, Madrid, Spain

[2] Départment de Biologie Comupationnelle, Institut Pasteur, Paris, France

[3] University of California, Davis, Davis, California, United States

[4] Harvard University, Boston, Massachusetts, United States

[5] California Institute of Technology, Pasadena, California, United States

[6] University of Maryland, College Park, MD, United States

Corresponding Author: Daniel Garijo
Email address: daniel.garijo@upm.es

Scientific software registries and repositories improve software findability and research transparency, provide information for software citations, and foster preservation of computational methods in a wide range of disciplines. Registries and repositories play a critical role by supporting research reproducibility and replicability, but developing them takes effort and few guidelines are available to help prospective creators of these resources. To address this need, the FORCE11 Software Citation Implementation Working Group convened a Task Force to distill the experiences of the managers of existing resources in setting expectations for all stakeholders. In this paper, we describe the resultant best practices which include defining the scope, policies, and rules that govern individual registries and repositories, along with the background, examples, and collaborative work that went into their development. We believe that establishing specific policies such as those presented here will help other scientific software registries and repositories better serve their users and their disciplines.

Nine Best Practices for Research Software Registries and Repositories

Daniel Garijo[1], Hervé Ménager[2], Lorraine Hwang[3], Ana Trisovic[4], Michael Hucka[5], Thomas Morrell[6], Alice Allen[7], Task Force on Best Practices for Software Registries[8], and SciCodes Consortium[9]

[1]**Universidad Politécnica de Madrid, Spain; daniel.garijo@upm.es**
[2]**Hub de Bioinformatique et Biostatistique – Département Biologie Computationnelle, Institut Pasteur, Université de Paris, Paris, France; herve.menager@pasteur.fr**
[3]**University of California, Davis, CA, United States; ljhwang@ucdavis.edu**
[4]**Harvard University, Cambridge, MA, United States; anatrisovic@g.harvard.edu**
[5]**California Institute of Technology, Pasadena, CA, United States; mhucka@library.caltech.edu**
[6]**California Institute of Technology, Pasadena, CA, United States; tmorrell@library.caltech.edu**
[7]**University of Maryland, College Park, MD, United States; aallen@ascl.net**
[8]**FORCE11 Software Citation Implementation Working Group**
[9]**Consortium of Scientific Software Registries and Repositories**

ABSTRACT

Scientific software registries and repositories improve software findability and research transparency, provide information for software citations, and foster preservation of computational methods in a wide range of disciplines. Registries and repositories play a critical role by supporting research reproducibility and replicability, but developing them takes effort and few guidelines are available to help prospective creators of these resources. To address this need, the *FORCE11 Software Citation Implementation Working Group* convened a Task Force to distill the experiences of the managers of existing resources in setting expectations for all stakeholders. In this paper, we describe the resultant best practices which include defining the scope, policies, and rules that govern individual registries and repositories, along with the background, examples, and collaborative work that went into their development. We believe that establishing specific policies such as those presented here will help other scientific software registries and repositories better serve their users and their disciplines.

1 INTRODUCTION

Research software is an essential constituent in scientific investigations (Wilson et al., 2014; Momcheva and Tollerud, 2015; Hettrick, 2018; Lamprecht et al., 2020), as it is often used to transform and prepare data, perform novel analyses on data, automate manual processes, and visualize results reported in scientific publications (Howison and Herbsleb, 2011). Research software is thus crucial for reproducibility and has been recognized by the scientific community as a research product in its own right—

one that should be properly described, accessible, and credited by others (Smith et al., 2016; Chue Hong et al., 2021). As a result of the increasing importance of computational methods, communities such as Research Data Alliance (RDA) (Berman and Crosas, 2020)[1] and FORCE11 (Bourne et al., 2012)[2] emerged to enable collaboration and establish best practices. Numerous software services that enable open community development of and access to research source code, such as GitHub[3] and Gitlab,[4] appeared and found a role in science. General-purpose repositories, such as Zenodo (CERN and OpenAIRE, 2013) and FigShare (Thelwall and Kousha, 2016), have expanded their scope beyond data to include software, and new repositories, such as Software Heritage (Di Cosmo and Zacchiroli, 2017), have been developed specifically for software. A large number of domain-specific research software registries and repositories have emerged for different scientific disciplines to ensure dissemination and reuse among their communities (Gentleman et al., 2004; Peckham et al., 2013; Greuel and Sperber, 2014; Allen and Schmidt, 2015; Gil et al., 2015, 2016).

Research software registries are typically indexes or catalogs of software metadata, without any code stored in them; while in *research software repositories*, software is both indexed *and* stored (Lamprecht et al., 2020). Both types of resource improve software discoverability and research transparency, provide information for software citations, and foster preservation of computational methods that might otherwise be lost over time, thereby supporting research reproducibility and replicability. Many provide or are integrated with other services, including indexing and archival services, that can be leveraged by librarians, digital archivists, journal editors and publishers, and researchers alike.

Transparency of the processes under which registries and repositories operate helps build trust with their user communities (Yakel et al., 2013; Frank et al., 2017). However, many domain research software resources have been developed independently, and thus policies amongst such resources are often heterogeneous and some may be omitted. Having specific policies in place ensures that users and administrators have reference documents to help define a shared understanding of the scope, practices, and rules that govern these resources.

Though recommendations and best practices for many aspects of science have been developed, no best practices existed that addressed the operations of software registries and repositories. To address this need, a Best Practices for Software Registries Task Force was proposed in June 2018 to the *FORCE11 Software Citation Implementation Working Group (SCIWG)*.[5] In seeking to improve the services software resources provide, software repository maintainers came together to learn from each other and promote interoperability. Both common practices and missing practices unfolded in these exchanges. These practices led to the development of nine best practices that set expectations for both users and maintainers of the resource by defining

---

[1] https://www.rd-alliance.org/
[2] https://www.force11.org/
[3] https://github.com/
[4] https://gitlab.com
[5] https://github.com/force11/force11-sciwg

79 management of its contents and allowed usages as well as clarifying positions on
80 sensitive issues such as attribution.

81 In this paper, we expand on our pre-print "Nine Best Practices for Research Software
82 Registries and Repositories: A Concise Guide" (Task Force on Best Practices for
83 Software Registries et al., 2020) to describe our best practices and their development.
84 Our guidelines are actionable, have a general purpose, and reflect the discussion of
85 a community of more than 30 experts who handle over 14 resources (registries or
86 repositories) across different scientific domains. Each guideline provides a rationale,
87 suggestions, and examples based on existing repositories or registries. To reduce
88 repetition, we refer to registries and repositories collectively as "resources."

89 The remainder of the paper is structured as follows. We first describe background and
90 related efforts in Section 2, followed by the methodology we used when structuring
91 the discussion for creating the guidelines (Section 3). We then describe the nine best
92 practices in Section 4, followed by a discussion (Section 5). Section 6 concludes the
93 paper by summarizing current efforts to continue the adoption of the proposed prac-
94 tices. Those who contributed to the development of this paper are listed in Appendix
95 A, and links to example policies are given in Appendix B. Appendix C provides up-
96 dated information about resources that have participated in crafting the best practices
97 and an overview of their main attributes.

2 BACKGROUND

99 In the last decade, much was written about a reproducibility crisis in science (Baker,
100 2016) stemming in large part from the lack of training in programming skills and the
101 unavailability of computational resources used in publications (Merali, 2010; Peng,
102 2011; Morin et al., 2012). On these grounds, national and international governments
103 have increased their interest in releasing artifacts of publicly-funded research to the
104 public (Office of Science and Technology Policy, 2016; Directorate-General for Re-
105 search and Innovation (European Commission), 2018; Australian Research Council,
106 2018; Chen et al., 2019; Ministère de l'Enseignement supérieur, de la Recherche et
107 de l'Innovation, 2021), and scientists have appealed to colleagues in their field to re-
108 lease software to improve research transparency (Weiner et al., 2009; Barnes, 2010;
109 Ince et al., 2012) and efficiency (Grosbol and Tody, 2010). Open Science initiatives
110 such as RDA and FORCE11 have emerged as a response to these calls for greater
111 transparency and reproducibility. Journals introduced policies encouraging (or even
112 requiring) that data and software be openly available to others (Editorial staff, 2019;
113 Fox et al., 2021). New tools have been developed to facilitate depositing research data
114 and software in a repository (Baruch, 2007; CERN and OpenAIRE, 2013; Di Cosmo
115 and Zacchiroli, 2017; Clyburne-Sherin et al., 2019; Brinckman et al., 2019; Trisovic
116 et al., 2020) and consequently, make them citable so authors and other contributors
117 gain recognition and credit for their work (Soito and Hwang, 2017; Du et al., 2021).

118 Support for disseminating research outputs has been proposed with FAIR and FAIR4RS
119 principles that state shared digital artifacts, such as data and software, should be Find-
120 able, Accessible, Interoperable, and Reusable (Wilkinson et al., 2016; Lamprecht
121 et al., 2020; Katz et al., 2021; Chue Hong et al., 2021). Conforming with the FAIR

principles for published software (Lamprecht et al., 2020) requires facilitating its discoverability, preferably in domain-specific resources (Jiménez et al., 2017). These resources should contain machine-readable metadata to improve the discoverability (Findable) and accessibility (Accessible) of research software through search engines or from within the resource itself. Furthering interoperability in FAIR is aided through the adoption of community standards *e.g.*, schema.org (Guha et al., 2016) or the ability to translate from one resource to another. The CodeMeta initiative (Jones et al., 2017) achieves this translation by creating a "Rosetta Stone" which maps the metadata terms used by each resource to a common schema. The CodeMeta schema[6] is an extension of schema.org which adds ten new fields to represent software-specific metadata. To date, CodeMeta has been adopted for representing software metadata by many repositories.[7,8]

As the usage of computational methods continues to grow, recommendations for improving research software have been proposed (Stodden et al., 2016) in many areas of science and software, as can be seen by the series of "Ten Simple Rules" articles offered by PLOS (Dashnow et al., 2014), sites such as AstroBetter,[9] courses to improve skills such as those offered by The Carpentries,[10] and attempts to measure the adoption of recognized best practices (Serban et al., 2020; Trisovic et al., 2022). Our quest for best practices complements these efforts by providing guides to the specific needs of research software registries and repositories.

3 METHODOLOGY

The best practices presented in this paper were developed by an international Task Force of the FORCE11 Software Citation Implementation Working Group (SCIWG). The Task Force was proposed in June 2018 by author Alice Allen, with the goal of developing a list of best practices for software registries and repositories. Working Group members and a broader group of managers of domain specific software resources formed the inaugural group. The resulting Task Force members were primarily managers and editors of resources from Europe, United States, and Australia. Due to the range in time zones, the Task Force held two meetings seven hours apart, with the expectation that, except for the meeting chair, participants would attend one of the two meetings. We generally refer to two meetings on the same day with the singular "meeting" in the discussions to follow.

The inaugural Task Force meeting (Feb, 2019) was attended by eighteen people representing fourteen different resources. Participants introduced themselves and provided some basic information about their resources, including repository name, starting year, number of records, and scope (discipline-specific or general purpose), as well as services provided by each resource (e.g., support of software citation, software deposits,

---

[6] https://codemeta.github.io/
[7] https://www.library.caltech.edu/caltechdata/news/enhanced-software-preservation-now-available-in-caltechdata
[8] https://hal.inria.fr/hal-01897934v3/codemeta
[9] https://www.astrobetter.com/
[10] https://carpentries.org/

Question	#Yes	#No	#Other
Is the resource discipline-specific?	6	8	0
Does the resource accept software only?	8	6	0
Does the resource require a software deposit?	2	12	0
Does the resource accept software deposits?	10	4	0
Can the resource mint DOIs?	6	8	0
Is the resource actively curated?	10	1	3
Can the resource be used to cite software?	11	2	1

Table 1. Overview of the information shared by the 14 resources which participated in the first Task Force meeting.

and DOI minting). Table 1 presents an overview of the collected responses, which highlight the efforts of the Task Force chairs to bring together both discipline-specific and general purpose resources. The "Other" category indicates that the answer needed clarifying text (e.g., for the question "is the repository actively curated?" some repositories are not manually curated, but have validation checks). Appendix C provides additional information on the questions asked to resource managers (Table C.1) and their responses (tables C.2, C.3 and C.4).

During the inaugural Task Force meeting, the chair laid out the goal of the Task Force, and the group was invited to brainstorm to identify commonalities for building a list of best practices. Participants also shared challenges they had faced in running their resources and policies they had enacted to manage these resources. The result of the brainstorming and discussion was a list of ideas collected in a common document.

Starting in May 2019 and continuing through the rest of 2019, the Task Force met on the third Thursday of each month and followed an iterative process to discuss, add to, and group ideas; refine and clarify the ideas into different practices, and define the practices more precisely. It was clear from the onset that, though our resources have goals in common, they are also very diverse and would be best served by best practices that were descriptive rather than prescriptive. We reached consensus on whether a practice should be a *best* practice through discussion and informal voting. Each best practice was given a title and a list of questions or needs that it addressed.

Our initial plan aimed at holding two Task Force meetings on the same day each month, in order to follow a common agenda with independent discussions built upon the previous month's meeting. However, the later meeting was often advantaged by the earlier discussion. For instance, if the early meeting developed a list of examples for one of the guidelines, the late meeting then refined and added to the list. Hence, discussions were only duplicated when needed, *e.g.*, where there was no consensus in the early group, and often proceeded in different directions according to the group's expertise and interest. Though we had not anticipated this, we found that holding two meetings each month on the same day accelerated the work, as work done in the second meeting of the day generally continued rather than repeating work done in the first meeting.

The resulting consensus from the meetings produced a list of the most broadly applicable practices, which became the initial list of best practices participants drew from during a two-day workshop, funded by the Sloan Foundation and held at the University of Maryland College Park, in November, 2019 (*Scientific Software Registry Collaboration Workshop*). A goal of the workshop was to develop the final recommendations on best practices for repositories and registries to the FORCE11 SCIWG. The workshop included participants outside the Task Force resulting in a broader set of contributions to the final list. In 2020, this group made additional refinements to the best practices during virtual meetings and through online collaborative writing producing in the guidelines described in the next section. The Task Force then transitioned into the SciCodes consortium.[11] SciCodes is a permanent community for research software registries and repositories with a particular focus on these best practices. SciCodes continued to collect information about involved registries and repositories, which are listed in Appendix C. We also include some analysis of the number of entries and date of creation of member resources. Appendix A lists the people who participated in these efforts.

4 BEST PRACTICES FOR REPOSITORIES AND REGISTRIES

Our recommendations are provided as nine separate policies or statements, each presented below with an explanation as to why we recommend the practice, what the practice describes, and specific considerations to take into account. The last paragraph of each best practice includes one or two examples and a link to Appendix B, which contains many examples from different registries and repositories.

These nine best practices, though not an exhaustive list, are applicable to the varied resources represented in the Task Force, so are likely to be broadly applicable to other scientific software repositories and registries. We believe that adopting these practices will help document, guide, and preserve these resources, and put them in a stronger position to serve their disciplines, users, and communities.[12]

4.1 Provide a public scope statement

The landscape of research software is diverse and complex due to the overlap between scientific domains, the variety of technical properties and environments, and the additional considerations resulting from funding, authors' affiliation, or intellectual property. A scope statement clarifies the type of software contained in the repository or indexed in the registry. Precisely defining a scope, therefore, helps those users of the resource who are looking for software to better understand the results they obtained.

Moreover, given that many of these resources accept submission of software packages, providing a precise and accessible definition will help researchers determine whether they should register or deposit software, and curators by making clear what is out of scope for the resource. Overall, a public scope manages the expectations of the potential depositor as well as the software seeker. It informs both what the resource does and does not contain.

---

[11]http://scicodes.net

[12]Please note that the information provided in this paper does not constitute legal advice.

The scope statement should describe:

- What is accepted, and acceptable, based on criteria covering scientific discipline, technical characteristics, and administrative properties

- What is not accepted, *i.e.* characteristics that preclude their incorporation in the resource

- Notable exceptions to these rules, if any

Particular criteria of relevance include the scientific community being served and the types of software listed in the registry or stored in the repository, such as source code, compiled executables, or software containers. The scope statement may also include criteria that must be satisfied by accepted software, such as whether certain software quality metrics must be fulfilled or whether a software project must be used in published research. Availability criteria can be considered, such as whether the code has to be publicly available, be in the public domain and/or have a license from a predefined set, or whether software registered in another registry or repository will be accepted.

An illustrating example of such a scope statement is the editorial policy[13] published by the Astrophysics Source Code Library (ASCL) (Allen et al., 2013), which states that it includes only software source code used in published astronomy and astrophysics research articles, and specifically excludes software available only as a binary or web service. Though the ASCL's focus is on research documented in peer-reviewed journals, its policy also explicitly states that it accepts source code used in successful theses. Other examples of scope statements can be found in Appendix B.

4.2 Provide guidance for users

Users accessing a resource to search for entries and browse or retrieve the description(s) of one or more software entries have to understand how to perform such actions. Although this guideline potentially applies to many public online resources, especially research databases, the potential complexity of the stored metadata and the curation mechanisms can seriously impede the understandability and usage of software registries and repositories.

User guidance material may include:

- How to perform common user tasks, such as searching the resource, or accessing the details of an entry

- Answers to questions that are often asked or can be anticipated, *e.g.*, with Frequently Asked Questions or tips and tricks pages

- Who to contact for questions or help

A separate section in these guidelines on the *Conditions of use policy* covers terms of use of the resource and how best to cite records in a resource and the resource itself.

---

[13]https://ascl.net/wordpress/submissions/editiorial-policy/

267 Guidance for users who wish to contribute software is covered in the next section,
268 *Provide guidance to software contributors*.

269 When writing guidelines for users, it is advisable to identify the types of users your
270 resource has or could potentially have and corresponding use cases. Guidance itself
271 should be offered in multiple forms, such as in-field prompts, linked explanations, and
272 completed examples. Any machine-readable access, such as an API, should be fully
273 described directly in the interface or by providing a pointer to existing documentation,
274 and should specify which formats are supported (*e.g.*, JSON-LD, XML) through
275 content negotiation if it is enabled.

276 Examples of such elements include, for instance, the bio.tools registry (Ison et al.,
277 2019) API user guide,[14] or the ORNL DAAC (ORNL, 2013) instructions for data
278 providers.[15] Additional examples of user guidance can be found in Appendix B.

4.3 Provide guidance to software contributors

280 Most software registries and repositories rely on a community model, whereby ex-
281 ternal contributors will provide software entries to the resource. The scope statement
282 will already have explained *what* is accepted and what is not; the contributor policy
283 addresses *who* can add or change software entries and the processes involved.

284 The contributor policy should therefore describe:

285   • Who can or cannot submit entries and/or metadata

286   • Required and optional metadata expected for deposited software

287   • Review process, if any

288   • Curation process, if any

289   • Procedures for updates (*e.g.*, who can do it, when it is done, how is it done)

290 Topics to consider when writing a contributor policy include whether the author(s) of
291 a software entry will be contacted if the contributor is not also an author and whether
292 contact is a condition or side-effect of the submission. Additionally, a contributor
293 policy should specify how persistent identifiers are assigned (if used) and should
294 state that depositors must comply with all applicable laws and not be intentionally
295 malicious.

296 Such material is provided in resources such as the Computational Infrastructure for
297 Geodynamics (Hwang and Kellogg, 2017) software contribution checklist[16] and the
298 CoMSES Net Computational Model Library (Janssen et al., 2008) model archival
299 tutorial.[17] Additional examples of guidance for software contributors can be found in
300 Appendix B.

---

[14]https://biotools.readthedocs.io/en/latest/api_usage_guide.html
[15]https://daac.ornl.gov/submit/
[16]https://geodynamics.org/cig/dev/code-donation/checklist/
[17]https://forum.comses.net/t/archiving-your-model-1-getting-started/7377

4.4 Establish an authorship policy

Because research software is often a research product, it is important to report authorship accurately, as it allows for proper scholarly credit and other types of attributions (Smith et al., 2016). However, even though authorship should be defined at the level of a given project, it can prove complicated to determine (Alliez et al., 2019). Roles in software development can widely vary as contributors change with time and versions, and contributions are difficult to gauge beyond the "commit," giving rise to complex situations. In this context, establishing a dedicated policy ensures that people are given due credit for their work. The policy also serves as a document that administrators can turn to in case disputes arise and allows proactive problem mitigation, rather than having to resort to reactive interpretation. Furthermore, having an authorship policy mirrors similar policies by journals and publishers and thus is part of a larger trend. Note that the authorship policy will be communicated at least partially to users through guidance provided to software contributors. Resource maintainers should ensure this policy remains consistent with the citation policies for the registry or repository (usually, the citation requirements for each piece of research software are under the authority of its owners).

The authorship policy should specify:

- How authorship is determined *e.g.*, a stated criteria by the contributors and/or the resource

- Policies around making changes to authorship

- The conflict resolution processes adopted to handle authorship disputes

When defining an authorship policy, resource maintainers should take into consideration whether those who are not coders, such as software testers or documentation maintainers, will be identified or credited as authors, as well as criteria for ordering the list of authors in cases of multiple authors, and how the resource handles large numbers of authors and group or consortium authorship. Resources may also include guidelines about how changes to authorship will be handled so each author receives proper credit for their contribution. Guidelines can help facilitate determining every contributors' role. In particular, the use of a credit vocabulary, such as the Contributor Roles Taxonomy (Allen et al., 2019), to describe authors' contributions should be considered for this purpose.[18]

An example of authorship policy is provided in the Ethics Guidelines[19] and the submission guide authorship section[20] of the *Journal of Open Source Software* (Katz et al., 2018), which provides rules for inclusion in the authors list. Additional examples of authorship policies can be found in Appendix B.

4.5 Document and share your metadata schema

The structure and semantics of the information stored in registries and repositories is sometimes complex, which can hinder the clarity, discovery, and reuse of the entries

---

[18]http://credit.niso.org/
[19]https://joss.theoj.org/about#ethics
[20]https://joss.readthedocs.io/en/latest/submitting.html#authorship

included in these resources. Publicly posting the metadata schema used for the entries helps individual and organizational users interested in a resource's information understand the structure and properties of the deposited information. The metadata structure helps to inform users how to interact with or ingest records in the resource. A metadata schema mapped to other schemas and an API specification can improve the interoperability between registries and repositories.

This practice should specify:

- The schema used and its version number. If a standard or community schema, such as *CodeMeta* (Jones et al., 2017) or *schema.org* (Guha et al., 2016) is used, the resource should reference its documentation or official website. If a custom schema is used, formal documentation such as a description of the schema and/or a data dictionary should be provided.

- Expected metadata when submitting software, including which fields are required and which are optional, and the format of the content in each field.

To improve the readability of the metadata schema and facilitate its translation to other standards, resources may provide a mapping (from the metadata schema used in the resource) to published standard schemas, through the form of a "cross-walk" (*e.g.*, the CodeMeta cross-walk[21]) and include an example entry from the repository that illustrates all the fields of the metadata schema. For instance, extensive documentation[22] is available for the biotoolsSchema (Ison et al., 2021) format, which is used in the bio.tools registry. Another example is the OntoSoft vocabulary,[23] used by the OntoSoft registry (Gil et al., 2015, 2016) and available in both machine-readable and human readable formats. Additional examples of metadata schemas can be found in Appendix B.

4.6 Stipulate conditions of use

The *conditions of use* document the terms under which users may use the contents provided by a website. In the case of software registries and repositories, these conditions should specifically state how the metadata regarding the entities of a resource can be used, attributed, and/or cited, and provide information about the licenses used for the code and binaries. This policy can forestall potential liabilities and difficulties that may arise, such as claims of damage for misinterpretation or misapplication of metadata. In addition, the conditions of use should clearly state how the metadata can and cannot be used, including for commercial purposes and in aggregate form.

This document should include:

- Legal disclaimers about the responsibility and liability borne by the registry or repository

- License and copyright information, both for individual entries and for the registry or repository as a whole

---

[21]https://codemeta.github.io/crosswalk/
[22]https://biotoolsschema.readthedocs.io/en/latest/
[23]http://ontosoft.org/software

378 • Conditions for the use of the metadata, including prohibitions, if any

379 • Preferred format for citing software entries

380 • Preferred format for attributing or citing the resource itself

381 When writing conditions of use, resource maintainers might consider what license
382 governs the metadata, if licensing requirements apply for findings and/or derivatives
383 of the resource, and whether there are differences in the terms and license for commer-
384 cial versus noncommercial use. Restrictions on the use of the metadata may also be
385 included, as well as a statement to the effect that the registry or repository makes no
386 guarantees about completeness and is not liable for any damages that could arise from
387 the use of the information. Technical restrictions, such as conditions of use of the API
388 (if one is available), may also be mentioned.

389 Conditions of use can be found for instance for DOE CODE (Ensor et al., 2017),
390 which in addition to the general conditions of use[24] specifies that the rules for usage
391 of the hosted code[25] are defined by their respective licenses. Additional examples of
392 conditions of use policies can be found in Appendix B.

4.7 State a privacy policy

394 Privacy policies define how personal data about users are stored, processed, exchanged
395 or removed. Having a privacy policy demonstrates a strong commitment to the privacy
396 of users of the registry or repository and allows the resource to comply with the legal
397 requirement of many countries in addition to those a home institution and/or funding
398 agencies may impose.

399 The privacy policy of a resource should describe:

400 • What information is collected and how long it is retained

401 • How the information, especially any personal data, is used

402 • Whether tracking is done, what is tracked, and how (*e.g.*, Google Analytics)

403 • Whether cookies are used

404 When writing a privacy policy, the specific personal data which are collected should
405 be detailed, as well as the justification for their resource, and whether these data are
406 sold and shared. Additionally, one should list explicitly the third-party tools used
407 to collect analytic information and potentially reference their privacy policies. If
408 users can receive emails as a result of visiting or downloading content, such potential
409 solicitations or notifications should be announced. Measures taken to protect users'
410 privacy and whether the resource complies with the *European Union Directive on*
411 *General Data Protection Regulation*[26] (GDPR) or other local laws, if applicable,
412 should be explained.[27] As a precaution, the statement can reserve the right to make

---

[24]https://www.osti.gov/disclaim
[25]https://www.osti.gov/doecode/faq#are-there-restrictions
[26]https://gdpr-info.eu/
[27]In the case of GDPR, the regulation applies to all European user personal data, even if the re-
source is not located in Europe.

413 changes to this privacy policy. Finally, a mechanism by which users can request the
414 removal of such information should be described.

415 For example, the SciCrunch's (Grethe et al., 2014) privacy policy[28] details what kind
416 of personal information is collected, how it is collected, and how it may be reused,
417 including by third-party websites through the use of cookies. Additional examples of
418 privacy policies can be found in Appendix B.

4.8 Provide a retention policy

420 Many software registries and repositories aim to facilitate the discovery and accessi-
421 bility of the objects they describe, *e.g.*, enabling search and citation, by making the
422 corresponding records permanently accessible. However, for various reasons, even in
423 such cases maintainers and curators may have to remove records. Common examples
424 include removing entries that are outdated, no longer meet the scope of the registry,
425 or are found to be in violation of policies. The resource should therefore document
426 retention goals and procedures so that users and depositors are aware of them.

427 The retention policy should describe:

428 • The length of time metadata and/or files are expected to be retained

429 • Under what conditions metadata and/or files are removed

430 • Who has the responsibility and ability to remove information

431 • Procedures to request that metadata and/or files be removed

432 The policy should take into account whether best practices for persistent identifiers
433 are followed, including resolvability, retention, and non-reuse of those identifiers. The
434 retention time provided by the resource should not be too prescriptive (*e.g.*, "for the
435 next 10 years"), but rather it should fit within the context of the underlying organiza-
436 tion(s) and its funding. This policy should also state who is allowed to edit metadata,
437 delete records, or delete files, and how these changes are performed to preserve the
438 broader consistency of the registry. Finally, the process by which data may be taken
439 offline and archived as well as the process for its possible retrieval should be thor-
440 oughly documented.

441 As an example, Bioconductor (Gentleman et al., 2004) has a deprecation process
442 through which software packages are removed if they cannot be successfully built or
443 tested, or upon specific request from the package maintainer. Their policy[29] specifies
444 who initiates this process and under which circumstances, as well as the successive
445 steps that lead to the removal of the package. Additional examples of retention poli-
446 cies can be found in Appendix B.

4.9 Disclose your end-of-life policy

448 Despite their usefulness, the long-term maintenance, sustainability, and persistence
449 of online scientific resources remains a challenge, and published web services or
450 databases can disappear after a few years (Veretnik et al., 2008; Kern et al., 2020).

---

[28]https://scicrunch.org/page/privacy
[29]https://bioconductor.org/developers/package-end-of-life/

451 Sharing a clear end-of-life policy increases trust in the community served by a registry
452 or repository. It demonstrates a thoughtful commitment to users by informing them
453 that provisions for the resource have been considered should the resource close or oth-
454 erwise end its services for its described artifacts. Such a policy sets expectations and
455 provides reassurance as to how long the records within the registry will be findable
456 and accessible in the future.

457 This policy should describe:

458   • Under what circumstances the resource might end its services

459   • What consequences would result from closure

460   • What will happen to the metadata and/or the software artifacts contained in the
461   resource in the event of closure

462   • If long-term preservation is expected, where metadata and/or software artifacts
463   will be migrated for preservation

464   • How a migration will be funded

465 Publishing an end-of-life policy is an opportunity to consider, in the event a resource
466 is closed, whether the records will remain available, and if so, how and for whom,
467 and under which conditions, such as archived status or "read-only." The restrictions
468 applicable to this policy, if any, should be considered and detailed. Establishing a
469 formal agreement or memorandum of understanding with another registry, repository,
470 or institution to receive and preserve the data or project, if applicable, might help to
471 prepare for such a liability.

472 Examples of such policies include the Zenodo end-of-life policy,[30] which states that
473 if Zenodo ceases its services, the data hosted in the resource will be migrated and the
474 DOIs provided would be updated to resolve to the new location (currently unspeci-
475 fied). Additional examples of end-of-life policies can be found in Appendix B.

476 A summary of the practices presented in this section can be found in Table 2.

5 DISCUSSION

478 The best practices described above serve as a guide for repositories and registries to
479 provide better service to their users, ranging from software developers and researchers
480 to publishers and search engines, and enable greater transparency about the operation
481 of their described resources. Implementing our practices provides users with signif-
482 icant information about *how* different resources operate, while preserving important
483 institutional knowledge, standardizing expectations, and guiding user interactions.

484 For instance, a public scope statement and guidance for users may directly impact
485 usability and, thus, the popularity of the repository. Resources including tools with a
486 simple design and unambiguous commands, as well as infographic guides or video
487 tutorials, ease the learning curve for new users. The guidance for software contribu-
488 tions, conditions of use, and sharing the metadata schema used may help eager users

---

[30]https://help.zenodo.org/

489 contribute new functionality or tools, which may also help in creating a community
490 around a resource. A privacy policy has become a requirement across geographic
491 boundaries and legal jurisdictions. An authorship policy is critical in facilitating col-
492 laborative work among researchers and minimizing the chances for disputes. Finally,
493 retention and end-of-life policies increase the trust and integrity of a repository ser-
494 vice.

Table 2. Summary of the best practices with recommendations and examples.

Practice, description and examples	Recommendations
1. Provide a public scope statement Informs both software depositor and resource seeker what the collection does and does not contain. **Example**: ASCL editorial policy.	• What is accepted, and acceptable, based on criteria covering scientific discipline, technical characteristics, and administrative properties • What is not accepted, i.e. characteristics that preclude their incorporation in the resource • Notable exceptions to these rules, if any
2. Provide guidance for users Helps users accessing a resource understand how to perform tasks like searching, browsing, and retrieving software entries. **Example**: bio.tools registry API user guide.	• How to perform common user tasks, like searching for collection, or accessing the details of an entry • Answers to questions that are often asked or can be anticipated • Point of contact for help and questions
3. Provide guidance to software contributors Specifies who can add or change software entries and explains the necessary processes. **Example**: Computational Infrastructure for Geodynamics contribution checklist.	• Who can or cannot submit entries and/or metadata • Required and optional metadata expected from software contributors • Procedures for updates, review process, curation process
4. Establish an authorship policy Ensures that contributors are given due credit for their work and to resolve disputes in case of conflict. **Example**: JOSS authorship policy.	• How authorship is determined e.g., a stated criteria by the contributors and/or the resource • Policies around making changes to authorship • Define the conflict resolution processes
5. Document and share your metadata schema Revealing the metadata schema used helps users understand the structure and properties of the deposited information. **Example**: OntoSoft vocabulary from the OntoSoft registry.	• Specify the used schema and its version number. Add reference to its documentation or official website. If a custom schema is used, provide documentation. • Expected metadata when submitting software

6. Stipulate conditions of use
Documents the terms under which users may use the provided resources, including metadata and software.
Example: DOE CODE acceptable use policy.

- Legal disclaimers about the responsibility and liability borne by the resource
- License and copyright information, both for individual entries and for the resource as a whole
- Conditions for the use of the metadata, including prohibitions, if any
- Preferred format for citing software entries; preferred format for attributing or citing the resource itself

7. State a privacy policy
Defines how personal data about users are stored, processed, exchanged, or removed.
Example: SciCrunch's privacy policy.

- What information is collected and how long it is retained
- How the information, especially any personal data, is used
- Whether tracking is done, what is tracked, and how; whether cookies are used

8. Provide a retention policy
Helps both users and depositors understand and anticipate retention goals and procedures.
Example: Bioconductor package deprecation.

- The length of time metadata and/or files are expected to be retained
- Under what conditions metadata and/or files are removed
- Who has the responsibility and ability to remove information; procedures to request that metadata and/or files be removed

9. Disclose end-of-life policy
Informs both users and depositors of how long the records within the resource will be findable and accessible in the future.
Example: Zenodo end-of-life policy.

- Circumstances under which the resource might end its services
- What consequences would result from closure
- What will happen to the metadata and/or the software artifacts contained in the resource in the event of closure
- If long-term preservation is expected, where metadata and/or software artifacts will be migrated for preservation; how a migration will be funded

Policies affecting a single community or domain were deliberately omitted when developing the best practices. First, an exhaustive list would have been a barrier to adoption and not applicable to every repository since each has a different perspective, audience, and motivation that drives policy development for their organization. Second, best practices that regulate the content of a resource are typically domain-specific to the artifact and left to resources to stipulate based on their needs. Participants in the 2019 Scientific Software Registry Collaboration Workshop were surprised to find that only four metadata elements were shared by all represented resources.[31] The diversity of our resources precludes prescriptive requirements, such as requiring specific metadata for records, so these were also deliberately omitted in the proposed best practices.

Hence, we focused on broadly applicable practices considered important by various

---

[31]The elements were: software name, description, keywords, and URL.

resources. For example, amongst the participating registries and repositories, very few had codes of conduct that govern the behavior of community members. Codes of conduct are warranted if resources are run as part of a community, especially if comments and reviews are solicited for deposits. In contrast, a code of conduct would be less useful for resources whose primary purpose is to make software and software metadata available for reuse. However, this does not negate their importance and their inclusion as best practices in other arenas concerning software.

As noted by the FAIR4RS movement, software is different than data, motivating the need for a separate effort to address software resources (Lamprecht et al., 2020; Katz et al., 2016). Even so, there are some similarities, and our effort complements and aligns well with recent guidelines developed in parallel to increase the transparency, responsibility, user focus, sustainability, and technology of data repositories. For example, both the TRUST Principles (Lin et al., 2020) and CoreTrustSeal Requirements Standards and Board (2019)) call for a repository to provide information on its scope and list the terms of use of its metadata to be considered compliant with TRUST or CoreTrustSeal, which aligns with our practices "*Provide a public scope statement*" and "*Stipulate conditions of use*". CoreTrustSeal and TRUST also require that a repository consider continuity of access, which we have expressed as the practice to "*Disclosing your end-of-life policy*". Our best practices differ in that they do not address, for example, staffing needs nor professional development for staff, as CoreTrustSeal requires, nor do our practices address protections against cyber or physical security threats, as the TRUST principles suggest. Inward-facing policies, such as documenting internal workflows and practices, are generally good in reducing operational risks, but internal management practices were considered out of scope of our guidelines.

Figure 1 shows the number of resources that support (partially or in their totality) each best practice. Though we see the proposed best practices as critical, many of the repositories that have actively participated in the discussions (14 resources in total) have yet to implement every one of them. We have observed that the first three practices (providing public scope statement, add guidance for users and for software contributors) have the widest adoption, while the retention, end-of-life, and authorship policy the least. Understanding the lag in the implementation across all of the best practices requires further engagement with the community.

Improving the adoption of our guidelines is one of the goals of SciCodes,[32] a recent consortium of scientific software registries and repositories. SciCodes evolved from the Task Force as a permanent community to continue the dialogue and share information between domains, including sharing of tools and ideas. SciCodes has also prioritized improving software citation (complementary to the efforts of the FORCE11 SCIWG) and tracking the impact of metadata and interoperability. In addition, SciCodes aims to understand barriers to implementing policies, ensure consistency between various best practices, and continue advocacy for software support by continuing dialogue between registries, repositories, researchers, and other stakeholders.

---

[32] http://scicodes.net
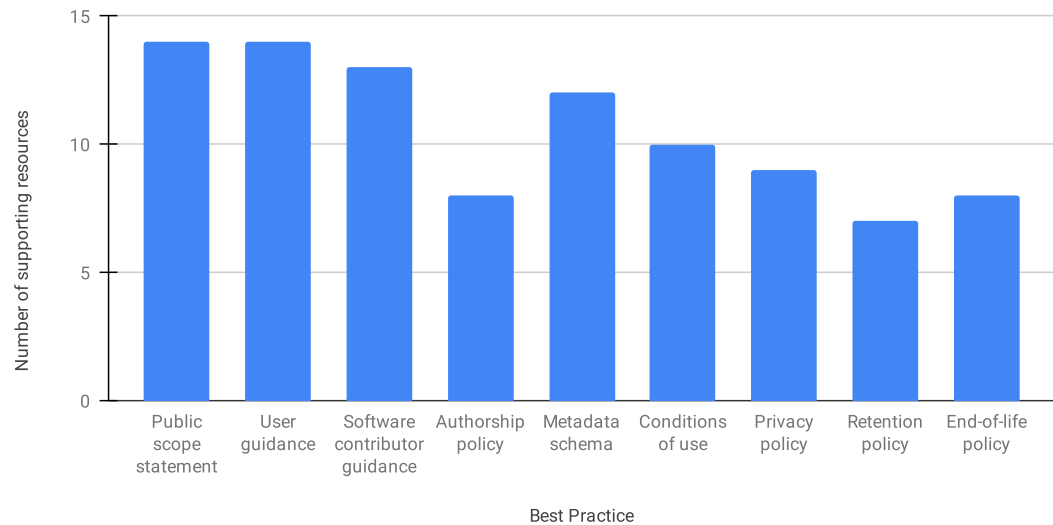
Figure 1. Number of resources supporting each best practice, out of 14 resources.

6 CONCLUSIONS

The dissemination and preservation of research material, where repositories and registries play a key role, lies at the heart of scientific advancement. This paper introduces nine best practices for research software registries and repositories. The practices are an outcome of a Task Force of the FORCE11 Software Citation Implementation Working Group and reflect the discussion, collaborative experiences, and consensus of over 30 experts and 14 resources.

The best practices are non-prescriptive, broadly applicable, and include examples and guidelines for their adoption by a community. They specify establishing the working domain (scope) and guidance for both users and software contributors, address legal concerns with privacy, use, and authorship policies, enhance usability by encouraging metadata sharing, and set expectations with retention and end-of-life policies. However, we believe additional work is needed to raise awareness and adoption across resources from different scientific disciplines. Through the SciCodes consortium, our goal is to continue implementing these practices more uniformly in our own registries and repositories and reduce the burdens of adoption. In addition to completing the adoption of these best practices, SciCodes will address topics such as tracking the impact of good metadata, improving interoperability between registries, and making our metadata discoverable by search engines and services such as Google Scholar, ORCID, and discipline indexers.

ACKNOWLEDGEMENTS

The best practices presented here were proposed and developed by a Task Force of the FORCE11 Software Citation Implementation Working Group. The following authors, randomly ordered, contributed equally to discussion, conceptualization, writing, reviewing, and editing this paper: Daniel Garijo, Lorraine Hwang, Hervé Ménager,

573 Alice Allen, Michael Hucka, Thomas Morrell, and Ana Trisovic.

574 **Task Force on Best Practices for Software Registries participants**: Alain Mon-
575 teil, Alejandra Gonzalez-Beltran, Alexandros Ioannidis, Alice Allen, Allen Lee, An-
576 dre Jackson, Bryce Mecum,Caifan Du, Carly Robinson, Daniel Garijo, Daniel Katz,
577 Genevieve Milliken, Hervé Ménager, Jurriaan Spaaks, Katrina Fenlon, Kristin Van-
578 derbilt, Lorraine Hwang, Michael Hucka, Neil Chue Hong, P. Wesley Ryan, Peter
579 Teuben, Shelley Stall, Stephan Druskat, Ted Carnevale, Thomas Morrell.

580 **SciCodes Consortium participants**: Alain Monteil, Alejandra Gonzalez-Beltran,
581 Alexandros Ioannidis, Alice Allen, Allen Lee, Ana Trisovic, Anita Bandrowski, Bruce
582 Wilson, Bryce Mecum, Carly Robinson, Celine Sarr, Colin Smith, Daniel Garijo,
583 David Long, Harry Bhadeshia, Hervé Ménager, Jeanette M. Sperhac, Joy Ku, Jurriaan
584 Spaaks, Kristin Vanderbilt, Lorraine Hwang, Matt Jones, Mercè Crosas, Michael R.
585 Crusoe, Mike Hucka, Ming Fang Wu, Morane Gruenpeter, Moritz Schubotz, Olaf
586 Teschke, Pete Meyer, Peter Teuben, Piotr Sliz, Sara Studwell, Shelley Stall, Ted
587 Carnevale, Tom Morrell, Tom Pollard, Wolfram Sperber.

REFERENCES

589 Allen, A., DuPrie, K., Berriman, B., Hanisch, R. J., Mink, J., and Teuben, P. J. (2013).
590   Astrophysics source code library. *Astronomical Data Analysis Software and Systems*
591   *XXII*, 475:387.
592 Allen, A. and Schmidt, J. (2015). Looking before leaping: Creating a software registry.
593   *Journal of Open Research Software, 3 (1):e15*, 3.
594 Allen, L., O'Connell, A., and Kiermer, V. (2019). How can we ensure visibility and
595   diversity in research contributions? how the contributor role taxonomy (credit) is
596   helping the shift from authorship to contributorship. *Learned Publishing*, 32(1):71–
597   74.
598 Alliez, P., Di Cosmo, R., Guedj, B., Girault, A., Hacid, M.-S., Legrand, A., and
599   Rougier, N. P. (2019). Attributing and referencing (research) software: Best practices
600   and outlook from inria. *Computing in Science and Engineering*, pages 1–14.
601 Australian Research Council (2018).   ARC open access policy.
602   https://www.arc.gov.au/policies-strategies/policy/arc-open-access-policy.
603 Baker, M. (2016).   1,500 scientists lift the lid on reproducibility.   *Nature News*,
604   533(7604):452.
605 Barnes, N. (2010). Publish your computer code: it is good enough. *Nature*, 467(753).
606 Baruch, P. (2007). Open access developments in france: the HAL open archives system.
607   *Learned publishing*, 20(4):267–282.
608 Berman, F. and Crosas, M. (2020). The research data alliance: Benefits and challenges
609   of building a community organization. *Harvard Data Science Review*, 2(1).
610 Bourne, P. E., Clark, T. W., Dale, R., de Waard, A., Herman, I., Hovy, E. H., and
611   Shotton, D. (2012).   Improving The Future of Research Communications and
612   e-Scholarship (Dagstuhl Perspectives Workshop 11331). *Dagstuhl Manifestos*,
613   1(1):41–60.
614 Brinckman, A., Chard, K., Gaffney, N., Hategan, M., Jones, M. B., Kowalik, K., Ku-
615   lasekaran, S., Ludäscher, B., Mecum, B. D., Nabrzyski, J., et al. (2019). Computing

616   environments for reproducibility: Capturing the "whole tale". *Future Generation*
617   *Computer Systems*, 94:854–867.

618 CERN and OpenAIRE (2013). Zenodo. https://doi.org/10.25495/7gxk-rd71.

619 Chen, X., Dallmeier-Tiessen, S., Dasler, R., Feger, S., Fokianos, P., Gonzalez, J. B.,
620   Hirvonsalo, H., Kousidis, D., Lavasa, A., Mele, S., Rodriguez, D. R., Šimko, T.,
621   Smith, T., Trisovic, A., Trzcinska, A., Tsanaktsidis, I., Zimmermann, M., Cranmer,
622   K., Heinrich, L., Watts, G., Hildreth, M., Lloret Iglesias, L., Lassila-Perini, K., and
623   Neubert, S. (2019). Open is not enough. *Nature Physics*, 15(2):113–119.

624 Chue Hong, N. P., Katz, D. S., Barker, M., Lamprecht, A.-L., Martinez, C., Pso-
625   mopoulos, F. E., Harrow, J., Castro, L. J., Gruenpeter, M., Martinez, P. A., and
626   Honeyman, T. (2021). FAIR principles for research software (FAIR4RS principles).
627   *Research Data Alliance*.

628 Clyburne-Sherin, A., Fei, X., and Green, S. A. (2019). Computational reproducibility
629   via containers in psychology. *Meta-psychology*, 3.

630 Dashnow, H., Lonsdale, A., and Bourne, P. E. (2014). Ten simple rules for writing a
631   PLOS ten simple rules article. *PLOS Computational Biology*, 10(10):1–5.

632 Di Cosmo, R. and Zacchiroli, S. (2017). Software Heritage: Why and How to Preserve
633   Software Source Code. In *iPRES 2017 - 14th International Conference on Digital*
634   *Preservation*, pages 1–10, Kyoto, Japan.

635 Directorate-General for Research and Innovation (European Commission) (2018). *Turn-*
636   *ing FAIR into reality: final report and action plan from the European Commission*
637   *expert group on FAIR data*. Publications Office of the European Union.

638 Du, C., Cohoon, J., Priem, J., Piwowar, H., Meyer, C., and Howison, J. (2021). Citeas:
639   Better software through sociotechnical change for better software citation. *CSCW*
640   *'21 Companion*. https://doi.org/10.1145/3462204.3482889.

641 Editorial staff (2019). Giving software its due. *Nature Methods*, 16(3):207. 00000.

642 Ensor, N., Stooksbury, S., Smith, A., Johnson, L. A., Vowell, L., Martin, M., Hensley,
643   M., Finkbeiner, D., Robinson, C., Knight, K., Nelson, J., Davis, L., Lee, I., Sherline,
644   C., Welsch, T., Billings, J. J., West, M. B., Sowers, T., and Watson, A. (2017). Doe
645   code. https://doi.org/10.11578/dc.20171031.3.

646 Fox, P., Erdmann, C., Stall, S., Griffies, S. M., Beal, L. M., Pinardi, N., Hanson, B.,
647   Friedrichs, M. A. M., Feakins, S., Bracco, A., Pirenne, B., and Legg, S. (2021).
648   Data and Software Sharing Guidance for Authors Submitting to AGU Journals.
649   We are thankful to Dr. Peter Fox and his initial draft contribution to this guidance
650   document while also saddened by his recent passing. Peter's contributions to the
651   community continue to have a great impact as represented in this document. Thank
652   you, Peter, you are missed.

653 Frank, R. D., Chen, Z., Crawford, E., Suzuka, K., and Yakel, E. (2017). Trust in
654   qualitative data repositories. *Proceedings of the Association for Information Science*
655   *and Technology*, 54(1):102–111.

656 Gentleman, R. C., Carey, V. J., Bates, D. M., Bolstad, B., Dettling, M., Dudoit, S.,
657   Ellis, B., Gautier, L., Ge, Y., Gentry, J., et al. (2004). Bioconductor: open software
658   development for computational biology and bioinformatics. *Genome biology*,
659   5(10):1–16.

660 Gil, Y., Garijo, D., Mishra, S., and Ratnakar, V. (2016). OntoSoft: A distributed
661   semantic registry for scientific software. In *2016 IEEE 12th International Conference*

662   *on e-Science (e-Science)*, pages 331–336.

663 Gil, Y., Ratnakar, V., and Garijo, D. (2015). OntoSoft: Capturing Scientific Software
664   Metadata. pages 1–4. ACM Press.

665 Grethe, J. S., Bandrowski, A., Banks, D., Condit, C., Gupta, A., Larson, S., Li, Y.,
666   Ozyurt, I., Stagg, A., Whetzel, P., et al. (2014). SciCrunch: a cooperative and
667   collaborative data and resource discovery platform for scientific communities. *Neu-*
668   *roinformatics*.

669 Greuel, G.-M. and Sperber, W. (2014). swMATH – an information service for mathe-
670   matical software. In Hong, H. and Yap, C., editors, *Mathematical Software – ICMS*
671   *2014*, pages 691–701, Berlin, Heidelberg. Springer Berlin Heidelberg.

672 Grosbol, P. and Tody, D. (2010). Making access to astronomical software more
673   efficient. *arXiv e-prints*, page arXiv:1004.4430.

674 Guha, R. V., Brickley, D., and Macbeth, S. (2016). Schema.org: Evolution of structured
675   data on the web. *Commun. ACM*, 59(2):44–51.

676 Hettrick, S. (2018).   Software in research survey.
677   https://doi.org/10.5281/zenodo.1183562.

678 Howison, J. and Herbsleb, J. D. (2011). Scientific software production: Incentives and
679   collaboration. In *Proceedings of the ACM 2011 Conference on Computer Supported*
680   *Cooperative Work*, CSCW '11, page 513–522, New York, NY, USA. Association
681   for Computing Machinery.

682 Hwang, L. and Kellogg, L. H. (2017). CIG Community Standards and Best Practices
683   for Scientific Software (Invited). SSA 2017 Annual Meeting Announcement and
684   Program. *Seismological Research Letters*, 88(2B):463–723.

685 Ince, D. C., Hatton, L., and Graham-Cumming, J. (2012). The case for open computer
686   programs. *Nature*, 482:485–488.

687 Ison, J., Ienasescu, H., Chmura, P., Rydza, E., Ménager, H., Kalaš, M., Schwämmle,
688   V., Grüning, B., Beard, N., Lopez, R., et al. (2019). The bio.tools registry of software
689   tools and data resources for the life sciences. *Genome biology*, 20(1):1–4.

690 Ison, J., Ienasescu, H., Rydza, E., Chmura, P., Rapacki, K., Gaignard, A., Schwämmle,
691   V., van Helden, J., Kalaš, M., and Ménager, H. (2021). biotoolsSchema: a formalized
692   schema for bioinformatics software description. *GigaScience*, 10(1):giaa157.

693 Janssen, M. A., Alessa, L. N., Barton, M., Bergin, S., and Lee, A. (2008). Towards a
694   community framework for agent-based modelling. *Journal of Artificial Societies and*
695   *Social Simulation*, 11(2):6.

696 Jiménez, R. C., Kuzak, M., Alhamdoosh, M., Barker, M., Batut, B., Borg, M., Capella-
697   Gutierrez, S., Chue Hong, N., Cook, M., Corpas, M., Flannery, M., Garcia, L.,
698   Gelpí, J. L., Gladman, S., Goble, C., González Ferreiro, M., Gonzalez-Beltran, A.,
699   Griffin, P. C., Grüning, B., Hagberg, J., Holub, P., Hooft, R., Ison, J., Katz, D. S.,
700   Leskošek, B., López Gómez, F., Oliveira, L. J., Mellor, D., Mosbergen, R., Mulder,
701   N., Perez-Riverol, Y., Pergl, R., Pichler, H., Pope, B., Sanz, F., Schneider, M. V.,
702   Stodden, V., Suchecki, R., Svobodová Vařeková, R., Talvik, H.-A., Todorov, I.,
703   Treloar, A., Tyagi, S., van Gompel, M., Vaughan, D., Via, A., Wang, X., Watson-
704   Haigh, N. S., and Crouch, S. (2017). Four simple recommendations to encourage
705   best practices in research software. *F1000Research*, 6:876.

706 Jones, M. B., Boettiger, C., Mayes, A. C., Slaughter, P., Gil, Y., Chue Hong, N., and
707   Goble, C. (2017). CodeMeta.

Katz, D. S., Gruenpeter, M., and Honeyman, T. (2021). Taking a fresh look at FAIR for research software. *F1000Research*, 2(5).

Katz, D. S., Niemeyer, K. E., and Smith, A. M. (2018). Publish your software: Introducing the journal of open source software (JOSS). *Computing in Science Engineering*, 20(3):84–88.

Katz, D. S., Niemeyer, K. E., Smith, A. M., Anderson, W. L., Boettiger, C., Hinsen, K., Hooft, R., Hucka, M., Lee, A., Löffler, F., Pollard, T., and Rios, F. (2016). Software vs. data in the context of citation.

Kern, F., Fehlmann, T., and Keller, A. (2020). On the lifetime of bioinformatics web services. *Nucleic Acids Research*, 48(22):12523–12533.

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. L., Chue Hong, N., Goble, C., and Capella-Gutierrez, S. (2020). Towards FAIR principles for research software. *Data Science*, 3(1):37–59.

Lin, D., Crabtree, J., Dillo, I., Downs, R. R., Edmunds, R., Giaretta, D., De Giusti, M., L'Hours, H., Hugo, W., Jenkyns, R., Khodiyar, V., Martone, M. E., Mokrane, M., Navale, V., Petters, J., Sierman, B., Sokolova, D. V., Stockhause, M., and Westbrook, J. (2020). The TRUST principles for digital repositories. 7(1):144.

Merali, Z. (2010). Computational science: ...error. *Nature*, 467(7317):775–777.

Ministère de l'Enseignement supérieur, de la Recherche et de l'Innovation (2021). Second national plan for open science. https://www.ouvrirlascience. fr/second-national-plan-for-open-science/.

Momcheva, I. and Tollerud, E. (2015). Software use in astronomy: an informal survey. *arXiv e-prints*, page arXiv:1507.03989.

Morin, A., Urban, J., Adams, P. D., Foster, I., Sali, A., Baker, D., and Sliz, P. (2012). Shining light into black boxes. *Science*, 336:159.

Office of Science and Technology Policy (2016). Principles for promoting access to federal government-supported scientific data and research findings through international scientific cooperation.

ORNL (2013). Oak Ridge National Laboratory Distributed Active Archive Center.

Peckham, S. D., Hutton, E. W. H., and Norris, B. (2013). A component-based approach to integrated modeling in the geosciences: The design of CSDMS. *Computers and Geosciences*, 53.

Peng, R. D. (2011). Reproducible research in computational science. *Science*, 334(6060):1226–1227.

Serban, A., van der Blom, K., Hoos, H., and Visser, J. (2020). Adoption and effects of software engineering best practices in machine learning. In *Proceedings of the 14th ACM / IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, ESEM '20, New York, NY, USA. Association for Computing Machinery.

Smith, A. M., Katz, D. S., and Niemeyer, K. E. (2016). Software citation principles. *PeerJ Computer Science*, 2:e86.

Soito, L. and Hwang, L. J. (2017). Citations for Software: Providing Identification, Access and Recognition for Research Software. *International Journal of Digital Curation*, 11(2):48–63. 00000.

Standards, C. and Board, C. (2019). CoreTrustSeal Trustworthy Data Repositories Requirements 2020–2022.

Stodden, V., McNutt, M., Bailey, D. H., Deelman, E., Gil, Y., Hanson, B., Heroux, M. A., Ioannidis, J. P. A., and Taufer, M. (2016). Enhancing reproducibility for computational methods. *Science*, 354:1240–1241.

Task Force on Best Practices for Software Registries, ., Monteil, A., Gonzalez-Beltran, A., Ioannidis, A., Allen, A., Lee, A., Bandrowski, A., Wilson, B. E., Mecum, B., Du, C. F., Robinson, C., Garijo, D., Katz, D. S., Long, D., Milliken, G., Ménager, H., Hausman, J., Spaaks, J. H., Fenlon, K., Vanderbilt, K., Hwang, L., Davis, L., Fenner, M., Crusoe, M. R., Hucka, M., Wu, M., Chue Hong, N., Teuben, P., Stall, S., Druskat, S., Carnevale, T., and Morrell, T. (2020). Nine Best Practices for Research Software Registries and Repositories: A Concise Guide. *arXiv e-prints*, page arXiv:2012.13117.

Thelwall, M. and Kousha, K. (2016). Figshare: a universal repository for academic resource sharing? *Online Information Review*, 40(3):333–346.

Trisovic, A., Durbin, P., Schlatter, T., Durand, G., Barbosa, S., Brooke, D., and Crosas, M. (2020). Advancing computational reproducibility in the dataverse data repository platform. In *Proceedings of the 3rd International Workshop on Practical Reproducible Evaluation of Computer Systems*, pages 15–20.

Trisovic, A., Lau, M. K., Pasquier, T., and Crosas, M. (2022). A large-scale study on research code quality and execution. *Scientific Data*, 9(1):60.

Veretnik, S., Fink, J. L., and Bourne, P. E. (2008). Computational biology resources lack persistence and usability. *PLoS computational biology*, 4(7):e1000136.

Weiner, B., Blanton, M. R., Coil, A. L., Cooper, M. C., Davé, R., Hogg, D. W., Holden, B. P., Jonsson, P., Kassin, S. A., Lotz, J. M., Moustakas, J., Newman, J. A., Prochaska, J. X., Teuben, P. J., Tremonti, C. A., and Willmer, C. N. A. (2009). Astronomical software wants to be free: A manifesto. In *astro2010: The Astronomy and Astrophysics Decadal Survey*, volume 2010, page P61.

Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., Gonzalez-Beltran, A., Gray, A. J. G., Groth, P., Goble, C., Grethe, J. S., Heringa, J., 't Hoen, P. A. C., Hooft, R., Kuhn, T., Kok, R., Kok, J., Lusher, S. J., Martone, M. E., Mons, A., Packer, A. L., Persson, B., Rocca-Serra, P., Roos, M., van Schaik, R., Sansone, S.-A., Schultes, E., Sengstag, T., Slater, T., Strawn, G., Swertz, M. A., Thompson, M., van der Lei, J., van Mulligen, E., Velterop, J., Waagmeester, A., Wittenburg, P., Wolstencroft, K., Zhao, J., and Mons, B. (2016). The FAIR guiding principles for scientific data management and stewardship. *Scientific Data*, 3(1):160018.

Wilson, G., Aruliah, D. A., Brown, C. T., Chue Hong, N. P., Davis, M., Guy, R. T., Haddock, S. H. D., Huff, K. D., Mitchell, I. M., Plumbley, M. D., Waugh, B., White, E. P., and Wilson, P. (2014). Best practices for scientific computing. *PLOS Biology*, 12(1):1–7.

Yakel, E., Faniel, I. M., Kriesberg, A., and Yoon, A. (2013). Trust in digital repositories. *International Journal of Digital Curation*, 8(1):143–156.

A CONTRIBUTORS

The following people contributed to the development of this paper through participation in the Best Practices Task Force meetings, 2019 Scientific Software Registry Collaboration Workshop, and/or SciCodes Consortium meetings:

Alain Monteil, Inria, HAL; Software Heritage

Alejandra Gonzalez-Beltran, Science and Technology Facilities Council, UK Research and Innovation, Science and Technology Facilities Council

Alexandros Ioannidis, CERN, Zenodo

Alice Allen, University of Maryland, College Park, Astrophysics Source Code Library

Allen Lee, Arizona State University, CoMSES Net Computational Model Library

Ana Trisovic, Harvard University, DataVerse

Anita Bandrowski, UCSD, SciCrunch

Bruce E. Wilson, Oak Ridge National Laboratory, ORNL Distributed Active Archive Center for Biogeochemical Dynamics

Bryce Mecum, NCEAS, UC Santa Barbara, CodeMeta

Caifan Du, iSchool, University of Texas at Austin, CiteAs

Carly Robinson, US Department of Energy, Office of Scientific and Technical Information, DOE CODE

Daniel Garijo, Universidad Politécnica de Madrid (formerly at Information Sciences Institute, University of Southern California), Ontosoft

Daniel S. Katz, University of Illinois at Urbana-Champaign, Associate EiC for JOSS, FORCE11 Software Citation Implementation Working Group, co-chair

David Long, Brigham Young University, IEEE GRS Remote Sensing Code Library

Genevieve Milliken, NYU Bobst Library, IASGE

Hervé Ménager, Hub de Bioinformatique et Biostatistique – Département Biologie Computationnelle, Institut Pasteur, ELIXIR bio.tools

Jessica Hausman, Jet Propulsion Laboratory, PO.DAAC

Jurriaan H. Spaaks, Netherlands eScience Center, Research Software Directory

Katrina Fenlon, University of Maryland, iSchool

Kristin Vanderbilt, Environmental Data Initiative, IMCR

Lorraine Hwang, University California Davis, Computational Infrastructure for Geodynamics

Lynn Davis, US Department of Energy, Office of Scientific and Technical Information, DOE CODE

Martin Fenner, Front Matter (formerly at DataCite), FORCE11 Software Citation Implementation Working Group, co-chair

Michael R. Crusoe, CWL, Debian-Med

Michael Hucka, California Institute of Technology, SBML; COMBINE

Mingfang Wu, Australian Research Data Commons, Australian Research Data Commons

Morane Gruenpeter, Inria, Software Heritage

Moritz Schubotz, FIZ Karlsruhe - Leibniz-Institute for Information Infrastructure, swMATH

Neil Chue Hong, Software Sustainability Institute / University of Edinburgh, Software Sustainability Institute; FORCE11 Software Citation Implementation Working Group,

844 co-chair

845 Pete Meyer, Harvard Medical School, SBGrid; BioGrids

846 Peter Teuben, University of Maryland, College Park, Astrophysics Source Code Li-
847 brary

848 Piotr Sliz, Harvard Medical School, SBGrid; BioGrids

849 Sara Studwell, US Department of Energy, Office of Scientific and Technical Informa-
850 tion, DOE CODE

851 Shelley Stall, American Geophysical Union, AGU Data Services

852 Stephan Druskat, German Aerospace Center (DLR)/University Jena/Humboldt-
853 Universität zu Berlin, Citation File Format

854 Ted Carnevale, Neuroscience Department, Yale University, ModelDB

855 Tom Morrell, Caltech Library, CaltechDATA

856 Tom Pollard, MIT / PhysioNet, PhysioNet

B POLICY EXAMPLES

857

Scope Statement

858

859 • Astrophysics Source Code Library. (n.d.). *Editorial policy*.
860   `https://ascl.net/wordpress/submissions/editiorial-`
861   `policy/`

862 • bio.tools. (n.d.). *Curators Guide*.
863   `https://biotools.readthedocs.io/en/latest/curators_`
864   `guide.html`

865 • Caltech Library. (2017). *Terms of Deposit*.
866   `https://data.caltech.edu/terms`

867 • Caltech Library. (2019). *CaltechDATA FAQ*.
868   `https://www.library.caltech.edu/caltechdata/faq`

869 • Computational Infrastructure for Geodynamics. (n.d.). *Code Donation*.
870   `https://geodynamics.org/cig/dev/code-donation/`

871 • CoMSES Net Computational Model Library. (n.d.). *Frequently Asked Ques-*
872   *tions*.
873   `https://www.comses.net/about/faq/#model-library`

874 • ORNL DAAC for Biogeochemical Dynamics. (n.d.). *Data Scope and Accep-*
875   *tance Policy*.
876   `https://daac.ornl.gov/submit/`

877 • RDA Registry and Research Data Australia. (2018). *Collection*. ARDC In-
878   tranet.
879   `https://intranet.ands.org.au/display/DOC/Collection`

880 • Remote Sensing Code Library. (n.d.). *Submit*.
881   `https://rscl-grss.org/submit.php`

882 • SciCrunch. (n.d.). *Curation Guide for SciCrunch Registry*.
883   `https://scicrunch.org/page/Curation%20Guidelines`

884 • U.S. Department of Energy: Office of Scientific and Technical Information.
885   (n.d.-a). *DOE CODE: Software Policy*. `https://www.osti.gov/doecode/`
886   `policy`

887 • U.S. Department of Energy: Office of Scientific and Technical Information.
888   (n.d.-b). *FAQs*. OSTI.GOV.
889   `https://www.osti.gov/faqs`

890 **Guidance for users**

891 • Astrophysics Source Code Library. (2021). *Q & A*
892   `https://ascl.net/home/getwp/898`

893 • bio.tools. (2021). *API Reference*
894   `https://biotools.readthedocs.io/en/latest/api_reference.`
895   `html`

896 • Caltech Library. (2019). *CaltechDATA FAQ*.
897   `https://www.library.caltech.edu/caltechdata/faq`

898 • Harvard Dataverse. (n.d.). *Curation and Data Management Services*
899   `https://support.dataverse.harvard.edu/curation-services`

900 • OntoSoft. (n.d.). *An Intelligent Assistant for Software Publication*
901   `https://ontosoft.org/users.html`

902 • ORNL DAAC for Biogeochemical Dynamics. (n.d.). *Learning*
903   `https://daac.ornl.gov/resources/learning/`

904 • U.S. Department of Energy: Office of Scientific and Technical Information.
905   (n.d.). *FAQs*. OSTI.GOV.
906   `https://www.osti.gov/doecode/faq`

907 **Guidance for software contributors**

908 • Astrophysics Source Code Library. (n.d.) *Submit a code*.
909   `https://ascl.net/code/submit`

910 • bio.tools. (n.d.) *Quick Start Guide*
911   `https://biotools.readthedocs.io/en/latest/quickstart_`
912   `guide.html`

913 • Computational Infrastructure for Geodynamics. *Contributing Software*
914   `https://geodynamics.org/cig/dev/code-donation/checklist/`

915 • CoMSES Net Computational Model Library (2019) *Archiving your model: 1.*
916   *Getting Started*
917   `https://forum.comses.net/t/archiving-your-model-1-`
918   `getting-started/7377`

919 • Harvard Dataverse. (n.d.) *For Journals*.
920   `https://support.dataverse.harvard.edu/journals`

Authorship

- Committee on Publication Ethics: COPE. (2020a). *Authorship and contributorship*.
  https://publicationethics.org/authorship

- Committee on Publication Ethics: COPE. (2020b). *Core practices*.
  https://publicationethics.org/core-practices

- Dagstuhl EAS Specification Draft. (2016). *The Software Credit Ontology*.
  https://dagstuhleas.github.io/SoftwareCreditRoles/
  doc/index-en.html#

- Journal of Open Source Software. (n.d.). *Ethics Guidelines*.
  https://joss.theoj.org/about#ethics

- ORNL DAAC (n.d) *Authorship Policy*.
  https://daac.ornl.gov/submit/

- PeerJ Journals. (n.d.-a). *Author Policies*.
  https://peerj.com/about/policies-and-procedures/#author-
  policies

- PeerJ Journals. (n.d.-b). *Publication Ethics*.
  https://peerj.com/about/policies-and-procedures/#publication-
  ethics

- PLOS ONE. (n.d.). *Authorship*.
  https://journals.plos.org/plosone/s/authorship

- National Center for Data to Health. (2019). The Contributor Role Ontology.
  https://github.com/data2health/contributor-role-ontology

Metadata Schema

- ANDS: Australian National Data Service. (n.d.). *Metadata*. ANDS.
  https://www.ands.org.au/working-with-data/metadata

- ANDS: Australian National Data Service. (2016). *ANDS Guide: Metadata*.
  https://www.ands.org.au/__data/assets/pdf_file/0004/
  728041/Metadata-Workinglevel.pdf

- Bernal, I. (2019). *Metadata for Data Repositories*.
  https://doi.org/10.5281/zenodo.3233486

- bio.tools. (2020). *Bio-tools/biotoolsSchema* [HTML].
  https://github.com/bio-tools/biotoolsSchema (Original work
  published 2015)

- bio.tools. (2019). *BiotoolsSchema documentation*.
  https://biotoolsschema.readthedocs.io/en/latest/

- The CodeMeta crosswalks. (n.d.)
  https://codemeta.github.io/crosswalk/

- Citation File Format (CFF). (n.d.)
  https://doi.org/10.5281/zenodo.1003149

- The DataVerse Project. (2020). DataVerse 4.0+ Metadata Crosswalk.
  https://docs.google.com/spreadsheets/d/10Luzti7svVTVKTA-px27oq3RxCUM-QbiTkm8iMd5C54

- OntoSoft. (2015). *OntoSoft Ontology*.
  https://ontosoft.org/ontology/software/

- Zenodo. (n.d.-a). *Schema for Depositing*.
  https://zenodo.org/schemas/records/record-v1.0.0.json

- Zenodo. (n.d.-b). *Schema for Published Record*.
  https://zenodo.org/schemas/deposits/records/legacyrecord.json

Conditions of use policy

- Allen Institute. (n.d.). *Terms of Use*.
  https://alleninstitute.org/legal/terms-use/

- Europeana. (n.d.). *Usage Guidelines for Metadata*. Europeana Collections.
  https://www.europeana.eu/portal/en/rights/metadata.html

- U.S. Department of Energy: Office of Scientific and Technical Information.
  (n.d.). *DOE CODE FAQ: Are there restrictions on the use of the material in DOE CODE?*
  https://www.osti.gov/doecode/faq#are-there-restrictions

- Zenodo. (n.d.). *Terms of Use*.
  https://about.zenodo.org/terms/

Privacy policy

- Allen Institute. (n.d.). *Privacy Policy*.
  https://alleninstitute.org/legal/privacy-policy/

- CoMSES Net. (n.d.). *Data Privacy Policy.*
  https://www.comses.net/about/data-privacy/

- Nature. (2020). *Privacy Policy*.
  https://www.nature.com/info/privacy

- Research Data Australia. (n.d.). *Privacy Policy*.
  https://researchdata.ands.org.au/page/privacy

- SciCrunch. (2018). *Privacy Policy*. SciCrunch.
  https://scicrunch.org/page/privacy

- Science Repository. (n.d.). *Privacy Policies*.
  https://www.sciencerepository.org/privacy

996   • Zenodo. (n.d.). *Privacy policy*.
997   https://about.zenodo.org/privacy-policy/

998 **Retention Policy**
999   • Bioconductor. (2020). *Package End of Life Policy*.
1000   https://bioconductor.org/developers/package-end-of-
1001   life/

1002   • Caltech Library. (n.d.). *CaltechDATA FAQ*.
1003   https://www.library.caltech.edu/caltechdata/faq

1004   • CoMSES Net Computational Model Library. (n.d.). *How long will models be*
1005   *stored in the Computational Model Library?*
1006   *https://www.comses.net/about/faq/*

1007   • Dryad. (2020). *Dryad FAQ - Publish and Preserve your Data*.
1008   https://datadryad.org/stash/faq#preserved

1009   • Software Heritage. (n.d.). *Content policy*.
1010   https://www.softwareheritage.org/legal/content-policy/

1011   • Zenodo. (n.d.). *General Policies v1.0*.
1012   https://about.zenodo.org/policies/

1013 **End-of-life policy**
1014   • Figshare. (n.d.). *Preservation and Continuity of Access Policy*.
1015   https://knowledge.figshare.com/articles/item/preservation-
1016   and-continuity-of-access-policy

1017   • Open Science Framework. (2019). *FAQs*. OSF Guides.
1018   http://help.osf.io/hc/en-us/articles/360019737894-
1019   FAQs

1020   • NASA Earth Science Data Preservation Content Specification (n.d.)
1021   https://earthdata.nasa.gov/esdis/eso/standards-and-
1022   references/preservation-content-spec

1023   • Zenodo. (n.d.). *Frequently Asked Questions*.
1024   https://help.zenodo.org/

C   RESOURCE INFORMATION

Since the first Task Force meeting was held in 2019, we have asked new resource representatives joining our community to provide the information shown in Table C.1. Thanks to this effort, the group has been able to learn about each resource, identify similarities and differences, and thus better inform our meeting discussions.

Table C.2, C.3 and C.4 provide an updated overview of the main features of all resources currently involved in the discussion and implementation of the best practices (30 resources in total as of December, 2021). Participating resources are diverse, and belong to a variety of discipline-specific (e.g. neurosciences, biology, geosciences,

₁₀₃₄ etc.) and domain generic repositories. Curated resources tend to have a lower number
₁₀₃₅ of software entries. Most resources have been created in the last 20 years, with the
₁₀₃₆ oldest resource dating from 1991. Most resources accept a software deposit, support
₁₀₃₇ DOIs to identify their entries, are actively curated, and can be used to cite software.

Question	Answer type
Repository name and abbreviation	Text
Repository home page	URL
Representative name and email address	Text
Is the repository discipline-specific?	Yes/No
Is the repository for discipline software only?	Yes/No
Is a software deposit accepted?	Yes/No
Is a software deposit required?	Yes/No
Does your resource have a public scope/editorial policy?	URL
Supported unique identifier(s) type(s)	Text
Can the repository mint DOIs?	Yes/No
Is the repository actively curated?	Yes/No/Other
How are entries added?	Text
Is your resource currently used to cite software?	Yes/No/Other
When did your resource start operating?	Year started
What is the number of records (as of filling date)?	Integer
Notes/comments/additional information	Text

Table C.1. Questions asked to resource representatives.

Question	#Yes	#No	#Other
Is the resource discipline-specific?	18	12	0
Does the resource accept software only?	17	13	0
Does the resource require a software deposit?	5	25	0
Does the resource accept a software deposit	22	8	0
Can the resource mint DOIs?	16	14	0
Is the resource actively curated?	21	3	6
Can the resource be used to cite software?	21	6	3

Table C.2. Information shared by 30 resources participating in the SciCodes consortium, as of December 2021

#Entries	#Resources
Unknown	2
Less than 1K	15
1K - 100K	10
More than 100K	3

Table C.3. Number of entries described in the resources of the SciCodes consortium, by December 2021

Creation year	#Resources
Before 2000	6
2000 - 2010	9
After 2010	15

Table C.4. Date of creation of the resources in the SciCodes consortium, by December 2021