2019 Scientific Software Registry Collaboration Workshop Report (draft)

Nov. 13-14, 2019 ♦ University of Maryland, College Park

This workshop was made possible by support from the Alfred P. Sloan Foundation; we are grateful for their support.

Image by Michael Hucka; used with permission

Table of Contents

Introduction

Computational methods have become increasingly important in research in many disciplines to the point that conducting research without software is very difficult or impossible [1], [2], yet the software that scientists write for their work is often not shared [3]–[5], as traditionally, there has been little reward for doing so [6]. The invisibility of this software threatens the transparency and reproducibility of science [7], [8], and much work has been undertaken in the past decade to encourage software release [9], [10], provide incentives for doing so, recognize software authors for their computational contributions to science through formal citations [11], [12], and advocate for research software to be treated as a first-class research object on par with published articles [13].

Altschul *et al.* [12] stated that "Good repositories for software and best practice workflows, especially if citable, would be a start" for improving research. Discipline- and institution-specific research software registries and repositories exist and are working with their communities of practice to improve the transparency and reproducibility of research in numerous ways. The focus of these resources may vary, but each offers expertise and practices that may benefit other similar endeavors.

Best practices

Guidelines for best practices for research software registries and repositories do not currently exist. Data repositories have well-developed standards, such as those required for Core Trust Seal[1] certification, which define best practices for registry procedure documentation, contingency planning, risk management, and other activities that are needed for successful long-term operation. As software registries have a different scope and different challenges, different best practices are required.

Software citation in research

Journals increasingly want software used in research to have a formal citation; this can be seen in editorials and author guidelines from broad scientific journals such as *Nature Methods* [14] and *Science,*[2] in domain-specific journals such as those published by the American Astronomical Society,[3] and from publishers handling many journals, such as Elsevier.[4] Though the FORCE11 Software Citation Principles [15] provide a roadmap for appropriate software citation, a number of unsolved challenges to implementation remain [16].

While software registries support making computational methods available and citable, they often do not make citation metadata for their entries available in a standard format. Several standards have been developed to capture the metadata needed for recognition and citation of

[1] https://www.coretrustseal.org
[2] https://www.sciencemag.org/authors/instructions-preparing-initial-manuscript
[3] https://journals.aas.org/data-guide/#source_codes
[4] https://www.elsevier.com/journals/astronomy-and-computing/2213-1337/guide-for-authors#71001

software, notably Codemeta [17] and Citation File Format (CFF) [18]. Despite the emergence of these standards, they have not been widely adopted by code registries nor by researchers. There are many barriers to educating scientists about software metadata and the role it plays in citation. Adoption of these standards is likely to progress more rapidly if software registries, which already contain the metadata necessary for citing software, provide Codemeta and/or CFF formatted files to software authors for editing and inclusion on their code download sites.

The FORCE11 Software Citation Implementation Working Group (SCIWG)[5] includes representatives from several research software resources, and a task force, led by the Astrophysics Source Code Library (ASCL),[6] was formed to improve communication between these and additional discipline-specific software registries and repositories. The initial goal was to collaborate on developing a list of best practices for such resources. More broadly, the aim is to discuss issues and challenges unique to running these services so that they might find common solutions, share ideas, and strengthen their resources and respective disciplines, individually and collectively. This task force has periodic conference calls in addition to ad hoc communications via email and other means.

Proposal

In July 2019, Michael Hucka, Thomas Morrell, and Stephen Davison from Caltech Library and Alice Allen from the ASCL submitted a funding proposal to the Alfred P. Sloan Foundation for a two-day workshop to focus on tackling some of the citation challenges for research software and to leverage the combined knowledge of these discipline-specific and institutional resources by gathering those who edit and maintain these services to share information and work on two specific projects to benefit our respective entities, our users, and our disciplines. The proposal was funded and the workshop was held at the University of Maryland, College Park on November 13-14, 2019.

This workshop is the first effort focused specifically on and for discipline-specific software registries and repositories. There is substantial benefit in sharing work methods, marketing ideas, and communication practices, as doing so can speed improvements to each of the respective services, making them more efficient and better able to meet the needs of their disciplines. The workshop goals were to demonstrate unique aspects of the respective services, discuss challenges and share solutions to common issues that arise in managing these resources, finalize a list of best practices for these and similar registries, and work cooperatively to speed adoption of the CodeMeta and/or CFF standards. A software developer knowledgeable about CodeMeta and CFF was available at the workshop and in the weeks following to assist those software services editors and managers who want to implement the production of these files for their holdings.

[5] https://github.com/force11/force11-sciwg
[6] http://ascl.net

Participants

Participation was by invitation. We worked for balance across disciplines and projects; we were particularly interested in keeping the focus of the workshop on the software registries, to serve their needs and start the work to build a community for those managing and editing these resources. We allowed one representative per discipline resource (with the exception of the two organizing entities, the Caltech Libraries and the ASCL) and ensured there were more registries represented than subject matter experts (SMEs) for various projects.

To build our invitation list, we considered those to who:

- Participated in one or more SCIWG Best Practices Task Force conference call
- Are subject matter experts in CFF, CodeMeta, and other projects related to scientific software and software citation
- Manage or otherwise work on similar resources (Software Heritage, swMATH, ModelDB, bio.tools), even if not participating in the SCIWG Best Practices Task Force
- Chair the SCIWG

Fourteen registry managers or editors representing twelve different registries attended, along with two general repository participants and six subject matter experts for a total of 22 in-person participants. See Table 1 for the list, affiliations, and registry or project names of the participants who were physically at the workshop.

Table 1: List of in-person participants

Alain Monteil, INRIA: HAL/Software Heritage
Alexandros Ioannidis, CERN: Zenodo
Alice Allen, University of Maryland: Astrophysics Source Code Library
Allen Lee, Arizona State University: CoMSES Net
Anita E Bandrowski, UCSD: SciCrunch Inc
Bryce Mecum, National Center for Ecological Analysis and Synthesis, UC Santa Barbara: CodeMeta
Caifan Du, University of Texas at Austin: CiteAs
Carly Robinson, DoE Office of Scientific and Technical Information: DOE CODE
Daniel Garijo, University of Southern California: OntoSoft
Daniel Katz, University of Illinois at Urbana-Champaign: FORCE11 Software Citation Implementation Working Group
Genevieve Milliken, New York University: Investigating & Archiving the Scholarly Git Experience (IASGE)
Hervé Ménager, Institut Pasteur: ELIXIR bio.tools
Jurriaan Spaaks, Netherlands eScience Center: Research Software Directory
Katrina Fenlon, University of Maryland: iSchool

Lorraine Hwang, UC Davis: Computational Infrastructure for Geodynamics
Michael Hucka, Caltech Library: Systems Biology Markup Language
P. Wesley Ryan: Astrophysics Source Code Library
Peter Teuben, University of Maryland: Astrophysics Source Code Library
Shelley Stall, American Geophysical Union: AGU Data Services
Stephan Druskat, German Aerospace Center (DLR)/University Jena/Humboldt-Universität zu Berlin: Citation File Format
Ted Carnevale, Neuroscience Department, Yale: ModelDB
Thomas Morrell, Caltech Library: CaltechDATA

36% women (8), 64% men (14). Total: 22

Remote access to workshop activities

We provided access to the fourteen plenary presentations and reports from the Best Practices breakout groups via Webex (umd.webex.com/meet/ascl). We also provided access to read and comment on the Google documents used in the Best Practices breakout groups as these documents were developed, and to other Google documents created during the workshop. We had one late cancellation and one no-show, both of whom took advantage of our remote option for viewing the presentations; one person also participated via Webex in one of the breakout groups. Table 2 shows those who attended remotely for some or all of what was available via Webex.

Table 2: List of Webex participants

Andre Jackson: Drug Disease Model Resources
Alejandra Gonzalez-Beltan: Science and Technology Facilities Council, UK Research and Innovation
Kristin Vanderbilt, Environmental Data Initiative: Information Management Code Registry
Neil Chue Hong, University of Edinburgh: Software Sustainability Institute

50% women (2), 50% men (2). Total: 4

An attendee from the ASCL was dedicated to scribing to capture the workshop presentations, discussions, and breakout reports.

Agenda

The agenda for the meeting included presentations by registry representatives and subject matter experts, several group activities and discussions, one pairs activity, and breakout groups for small-group work. The first day of the meeting was planned out; the second day was less scripted and allowed time for participants to determine what to focus on and how they wanted to work. The agenda in Table 3 reflects the selections the participants made during the workshop as to how to spend their time and efforts.

Table 3: Agenda

Wednesday, November 13
8:30 Arrival and coffee/tea
9:00 Introductions (What are your expectations?)
9:30 Presentations (available via Webex)
- Hervé Ménager, ELIXIR bio.tools
- Ted Carnevale, ModelDB
- Anita E Bandrowski, SciCrunch Inc
- Jurriaan Spaaks, Research Software Directory

10:20 Coffee/tea break
10:35 Best Practices for Research Software Registries breakout groups
- Repository Scope statement and Retention Policy
- Metadata Schema and Conditions of Use Policy
- End-of-life Policy
- Privacy Policy
- Ethics/Authorship Policy

11:45 Reports from breakout groups and status of Best Practices (available via Webex)
12:15 Lunch break
1:30 Presentations (available via Webex)
- Lorraine Hwang, Computational Infrastructure for Geodynamics
- Daniel Garijo, OntoSoft
- Caifan Du, CiteAs
- Allen Lee, CoMSES Net
- Alex Ioannidis, Zenodo

2:20 Coffee/tea break
2:40 Group discussion
- Shared challenges

What are they?
How might we work together to mitigate these?
- Individual solutions to share

What technique/process/habit/requirement has been successful for your resource?

What solutions already exist that we might leverage?
3:45 State of software citation amongst assembled resources
4:15 Overview of CodeMeta and Citation File Format (available via Webex)
Slides
- 4:15-4:25 Stephan Druskat, Citation File Format
- 4:25-4:35 Bryce Mecum, CodeMeta
- 4:35-5:15 Activities: Crosswalking our resources and Metadata Bingo
Discussion: Creating software citation files for software authors' use
5:15 Temperature taking (how's the workshop going?), recap, and overview of day 2 agenda
5:30 End session
7:00 Workshop dinner

Thursday, November 14
8:30 Arrival and coffee/tea
9:00 Presentations (available via Webex)
- Carly Robinson, DOE CODE: Software Services Platform and Search Tool
- Tom Morrell, CaltechDATA
- Alice Allen, Astrophysics Source Code Library
9:30 Discussion and breakout groups
- Developing CodeMeta files for software author use
- Software value metadata statement
- Virtual code registry
- Improving ASCL's generated CITATION.cff and codemeta.json files
10:30 Coffee/tea break
10:45 Temperature taking (how's the workshop going? Do we need to revisit or spend more time on something?)
11:00 Continuation of breakout groups
12:15 Lunch break
1:15 Reports from morning's breakout groups
2:20 Activity: Links to download data from various registries
2:30 Workshop photo and coffee/tea break
2:45 Breakout groups
- Improving CodeMeta documentation
- CFF/CodeMeta/Zenodo matchup: A workflow
- Registry guidance for users: Possible examples for User Guidance best practice
- Best practices documentation clean-up
4:30 Reports from breakout groups
5:00 Recap and next steps
5:20 Workshop improvement
5:30 End session

Presentations

Registry and repository managers were encouraged to give a 10-minute presentation on their resource, as were subject matter experts, to serve as introductions to the different projects represented. To keep the workshop on schedule, the time limit for these was enforced. The presentations were scheduled throughout the first day and the morning of the second day of the workshop.

Presentations by registry and repository editors

bio.tools and the ELIXIR Tools Platform, **Hervé Ménager** (slides, PDF)

> Bio.tools is a registry for bioinformatics software, which aims at providing a comprehensive coverage for software tools and data services across the spectrum of biological and biomedical sciences. This registry, which describes more than 14000 entries, provides a unique resource for scientists to find, understand, utilize and cite the software they need in their day-to-day work. The various resources are described in a rigorous semantics and syntax, providing end-users with the convenience of concise, consistent and therefore comparable information. Each bio.tools entry is assigned a human-readable, unique identifier based on the resource name, e.g. biotools:signalp. These identifiers provide a persistent reference to our "Tool Cards" of essential information, as well as a means to trace resources and integrate bio.tools data with other resources. All the bio.tools data and technical components are available under open license. bio.tools development is supported by ELIXIR - the European infrastructure for life science information.
>
> In order to open further the data maintained within bio.tools and integrate it with others maintained by the ELIXIR Tools platform and/or other institutions and communities, we are currently building a new "Tools ecosystem" (e.g. bio.tools, Bioconda, BioContainers, OpenEBench and Galaxy). This is building on top of the content that the ELIXIR Tools platform aggregated and curated over the last years.

ModelDB: a database of published computational neuroscience models, **Ted Carnevale** (slides, PDF)

> ModelDB is a curated online database of published models in the domain of computational neuroscience. Its design goals are to facilitate neuroscience research by promoting the discoverability, reproducibility, verifiability, understanding, and attributed re-use of such models. Its browser-based user interface simplifies the tasks of: entering new models into the database; finding, downloading, and running models of interest; exploring the experimental evidence on which models are based; and understanding what is actually included in a model. As of November 2019, it contains more than 1480 models implemented with more than 100 different programming languages or simulation environments, and has been cited by more than 600 scientific publications whose authors have obtained code from it and/or submitted their own code to it.

RRIDs, an emerging standard in journal publishing, and the tools that make RRIDs a reality, **Anita E Bandrowski** (slides, PDF)

The SciCrunch Registry holds metadata records that describe digital resources, e.g., software, databases, projects and also services. Most of these are produced as a result of government funding and are available to the scientific community. Resources are manually curated according to our curation policies to make sure the information is accurate. The registry is the source of RRIDs, persistent unique identifiers, that are used in over 18,000 journal articles as a means of citing research resources.

Research Software Directory, **Jurriaan Spaaks** (slides, PDF)

This presentation introduced the Research Software Directory, a content management system tailored to research software. The system has been designed to publish metadata about software packages as easily readable, well-structured web pages, such that web page visitors can quickly judge whether the software advertised in the content management system will help them address their problems. Furthermore, the system aims to minimize the burden that is placed on content providers, *i.e.* the developers of a given software package. This minimization is realized by harvesting whatever data is already available from other systems such as Zotero, GitHub, and Zenodo. Such automated integration helps with keeping the Research Software Directory contents up to date and accurate. The Research Software Directory uses recent advances in machine-readable citation metadata to disseminate software reference manager files in BibTex, EndNote, and RIS formats, while promoting discovery by search engines through the use of schema.org metadata embedded within individual web pages.

Computational Infrastructure for Geodynamics, **Lorraine Hwang** (slides, pptx)

The Computational Infrastructure for Geodynamics (CIG) is a community of practice for the development and dissemination of open source codes in geodynamics. Our community includes users, developers, and user-developers who we support through various activities including workshops and hackathons. CIG's software repository is open to codes within the domain who meet our minimum software best practices and as approved by governance. We leverage 3rd party tools for hosting our repository (GitHub) and to obtain a persistent identifier (Zenodo). We maintain software pages for all codes in which all released versions and binaries are available. For more information, see geodynamics.org.

From software metadata registries to knowledge graphs: OntoSoft and OKG-SOFT, **Daniel Garijo** (slides, PDF)

Scientific software is being increasingly recognized as an important asset for creating understandable and reproducible scientific products. Software registries are the entry point to finding, comparing and learning about scientific software. OntoSoft is a distributed semantic software metadata registry with hundreds of software entries. The key characteristic of this portal is that we separate the metadata for this software into six

different categories that are very close to how scientists understand concepts of software: Identify, Understand, Execute, Do research, Get support, Update. We collected this metadata from questions that the users have to answer. Once we have this it is very easy to do cross-comparison of different software that may be used for the same thing. What are the different entries, what do they do, what is their overall scope. A series of requirements for software reusability are making us move towards knowledge graphs of scientific software metadata.

CoMSES Net: Promoting reproducibility, transparency, and knowledge sharing in computational social ecological systems science, **Allen Lee** (slides, pptx)

Computational modeling is an increasingly critical tool in the study of social ecological systems which typically exhibit non-linear complex systems dynamics. These computational decision support tools assist in policy and planning, scenario development, environmental management, resource investment, and security and disaster preparedness and are a research keystone tool supporting the earth, environmental, and social sciences. To be more useful for science and policy it is critical that natural, social, and computer scientists work together to design maintainable and sustainable codebases and next generation modeling environments. The core team behind CoMSES Net, the Network for Computational Modeling in the Social and Ecological Sciences has been working to address these challenges and improve computational modeling practices for social ecological systems science since 2007 through cyberinfrastructure development and the establishment and promotion of community standards, guides to good practice, and partnerships with like-minded organizations. As such, the CoMSES Science Gateway (https://comses.net) provides information resources on modeling frameworks, journals, community events, educational resources, career opportunities, and the CoMSES Model Library. The CoMSES Model Library is a digital repository where researchers can publish and archive their computational models and associated documentation and metadata in accordance with FAIR and FORCE11 software citation principles and request can request peer review of their published computational models. Like journal reviews, model peer review is conducted by volunteer experts from our community who verify that the models can be run, are sufficiently and clearly documented, and have clean code.

DOE CODE: Software Services Platform and Search Tool, **Carly Robinson** (slides, pptx)

The US Department of Energy (DOE), Office of Scientific and Technical Information (OSTI), has responsibility for collecting, preserving, and disseminating the R&D results emanating from DOE funding. This includes scientific software and code. In 2017, OSTI embarked on the creation of a new software submission and dissemination tool. With input from 9 requirements teams, OSTI launched DOE CODE in November of 2017. DOE CODE is DOE's software services platform and discovery tool, developed open source, offering easy submission of software code and metadata, DOI assignment, repository services, and APIs for submission and discovery.

CaltechDATA, **Tom Morrell** (slides, pptx)

CaltechDATA is an institutional repository for data and software associated with Caltech. The repository is based on Invenio v3, and Caltech Library partners with the InvenioRDM project to create shared repository infrastructure for software and data repositories. CaltechDATA has almost 200 software records, most of which have been generated via a GitHub integration. Our repository uses CodeMeta to solve the issue of incomplete metadata available from GitHub, and has resulted in much richer software records. New repository features, including CodeMeta and Binder integration, have been successfully and efficiently developed outside the repository stack using the API.

Astrophysics Source Code Library, **Alice Allen** (slides, PDF)

The ASCL accepts (lower case) open source software used in astronomy research; it was started in 1999. Entries are created either by one of the ASCL's editors or by submission by a software author or other non-editor. Its metadata is kept deliberately light to keep the resource manageable, as other ASCL-like projects in astronomy proved to be unsustainable due in full or in part to lack of metadata maintenance. Because the ASCL dropped its requirement in 2012 that software be deposited with the ASCL as a condition of registering the code, the ASCL has been able to grow more quickly since then and make more research more transparent than it would have otherwise. The presentation ended with a demonstration of the ASCL's overlays for creating codemeta.json and CITATION.cff files to provide to software authors for editing and placement on their code's download site.

Presentations by subject matter experts

Zenodo and Software, **Alex Ioannidis** (slides, PDF)

Zenodo, a CERN-hosted repository for the long tail of science, has been continuously evolving for the past years to accommodate the evergrowing needs of the scientific community. By integrating with platforms like GitHub and introducing the concept of DOI versioning, Zenodo has played a major role in making software a first-class citizen in scholarly communication. To further pursue this cause and in collaboration with the NASA Astrophysics Data System and the American Astronomical Society, the Asclepias project has focused its efforts on the complex nature of software citation and provides the tooling and infrastructure to address it. In a similar spirit of promoting open and reusable infrastructure, Zenodo is also participating in the development of InvenioRDM, a turn-key solution for research data management repositories, allowing institutions to benefit from the same technology and features that power Zenodo. Looking into the future, a new "GitHub Actions"-powered workflow is explored to allow flexibility in the way software packages integrate with registries and repositories.

CiteAs, **Caifan Du** (slides, PDF)

CiteAs (http://citeas.org/) is an interactive search engine deliberately developed to bridge the gap between software end-users and creators in the implementation of software

citation. CiteAs allows Web search for any findable research product online with a proper identifier input, be it a DOI, a URL link to its repository or project page, or the product name. With a focus on research software products, CiteAs can return a formatted, downloadable, and customizable citation based on all the available metadata found about the research product being searched. CiteAs can identify and parse .CFF file, CodeMeta file, domain-specific citation file such as R Description file, and natural language requests that creators have made online. Once the information is identified and parsed, CiteAs extracts the necessary information from the found metadata to construct a formatted citation. The provenance of the citation metadata will be displayed on the CiteAs interface. We hope that by presenting this information, users will be motivated and informed of elements that may be absent or that they can otherwise improve to make their citation request more visible. CiteAs is still in continuous development. We developed a human-annotated dataset of software mentions in scientific publications and utilized it to train a machine learning module in the CiteAs system. Ultimately CiteAs will be able to take any research paper, detect any informal software mentions in it, link them to the right citation request of the software creator, and make better citation suggestions to software end-users. We expect it to be a useful two-sided system that incentivizes both research software end-users and creators to adopt better citation practice.

The Citation File Format (CFF): Why, what, how?, **Stephan Druskat** (slides, PDF)

Software citation has a metadata problem. Awareness of software's status as a valid research product, and that it should be cited like papers are cited, is growing, but the relevant metadata is often not readily available with the software itself, or it is unreliable. One solution is to provide citation metadata files (CITATION.cff) in the Citation File Format (CFF) with code. CFF is human- and machine-readable and -writable, self-descriptive, compatible with other formats such as CodeMeta and BibTeX, and developed on the Software Citation Principles. Available tooling includes a schema and a schema validation tool, a simple web form for manual creation and curation of metadata, a conversion tool, and build system plugins to auto-generate and complete CFF files. The format and its tooling allow downstream users, both human and machine (such as software registries), to use CITATION.cff to find, read, understand, convert and re-use citation-relevant software metadata.

The CodeMeta Project, **Bryce Mecum** (slides, PDF)

The CodeMeta project defines a minimal, common metadata language to describe science software and code, enabling standardized exchange of metadata across repositories and organizations. A vast number of software repositories exist today that describe their software holdings in similar but ultimately incompatible ways. To address this problem, the CodeMeta project created the CodeMeta Metadata Crosswalk which maps the terms from these many software repositories onto a set of common vocabulary terms thereby making it possible to understand and exchange software metadata across repositories. The CodeMeta project also provides a JSON-LD serialization suitable for

archiving alongside software or embedding in web contexts otherwise describing the software.

Discussions

Shared challenges and solutions to share

After the Day 1 afternoon break, a group discussion was started on shared challenges and what some of the resources were doing to try to meet these challenges.

The afternoon's discussion focused on citation, how software is different from data, versioning, and identifiers.

The wide-ranging discussion on identifiers started on identifying both specific versions of data or software, and for collections or concepts, wherein the resource (data or code) is cited but the citation is not to a specific version. Resources are interested in finding workable ways to create connections between a concept and a hierarchy or collection, especially when it is left up to authors or developers to decide what they want to assign a persistent identifier to. One issue is that there is little guidance and few examples provided to developers on how to do this well and thoughtfully to allow connections that show that a dataset is part of or related to another dataset.

This was likened to software; when evaluating different software packages for performing certain tasks, one would want to cite an identifier, such as a DOI, for the concept, rather than a specific version of the software. Another use case for concept or collection identifiers and having them connected to version identifiers is to help learn the impact of creating that tool or collecting that data. This is information a program officer may want to have, to learn who is using a tool that money was spent on.

One would want to cite the identifier for the specific version of a code used for a research result to aid reproducibility of that result.

Another focus of identifiers was the use of multiple identifiers for the same object, or portion of an object. For example, one software package may have an RRID, DOI, and ASCL ID assigned to it. The group discussed the use of meta-resolvers, such as n2t.net and Identifiers.org, but though these are able to resolve different identifiers, they do not tell the user that the object under consideration has other identifiers assigned to it. One participant suggested that Scholix (http://www.scholix.org/) might be able to provide other identifiers for the object, as it already links literature, for example, to data; it also has a relationship with Identifiers.org that allows it to resolve non-DOI identifiers.

State of software citation amongst assembled resources

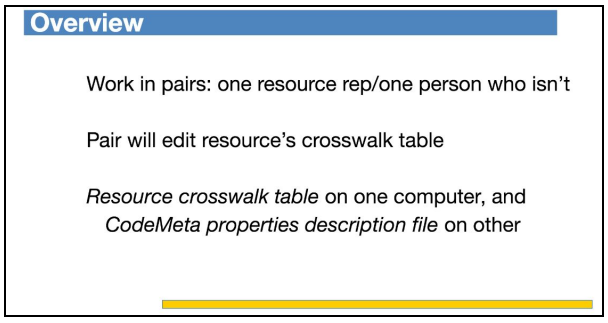
Participants discussed the state of software citation in their resources on the afternoon of the first day of the workshop. Though several of the software registries (such as ModelDB and ASCL) had metrics for formal software citations, most, including Software Heritage, Research Software Directory, CoMSES, and SBML, did not, as some services are relatively new, have been more focused on data, initially tracked other metrics such as software downloads, and/or have been struggling to find and offer good guidance regarding software citation. For example, AGU has recently been pushing both software and data citation for their 22 journals, with soft enforcement of citation to start on March 18, 2019 and hard enforcement on August 1, 2019. However, journal editors and reviewers still have many questions about software citation expectations. Shelley Stall of AGU has found guidance that has come out of the FORCE11 community and discussed even at this workshop very helpful and would be disseminated by AGU at its December 2019 meeting.

Several resources, such as CaltechDATA, HAL, ASCL, and OntoSoft, offer "recommended citation" text to their users or otherwise provide guidance on citing software, and others, such as bio.tools, expressed a desire to do so. There was general consensus that more could be done to improve and measure citation of scientific code within these resources. This discussion came just before the presentations on CodeMeta and CITATION.cff, deliberately so, as these standard file formats offer a way to improve software citation.

Activities

Crosswalking our resources

This activity's goal was to edit an existing table that had been set up for each of the registries/repositories in the workshop to crosswalk their data fields to two specific software metadata formats: Citation File Format and CodeMeta. These tables, once completed, would then be available to be added to the crosswalk tables on the CodeMeta site. Further, as one of the goals of the workshop was to inspire some subset of the resource managers to offer one or both of these software metadata files to their users, a crosswalk table would provide information needed for that task.



We anticipated that some participants would be unfamiliar with the CodeMeta and CITATION.cff projects, so scheduled this activity immediately after Stephan Druskat's presentation on CITATION.cff and Bryce Mecum's presentation on the CodeMeta project. We asked participants to pair up, with each registry manager working with someone who was not a registry manager. This would allow each team to have two laptops to work with, with one laptop used to edit the table, and the other to show the properties of each of the CodeMeta fields.

Results of this activity were mixed; participants learned more about each of these software metadata file formats, but struggled with completing the tables. Some of the registries have implicit fields that are hard to parse out, and even explicit data may not be easily categorized into a specific field. In addition, it may be hard to programmatically pull out the data needed to create one of these software files. Still, several tables were developed enough to be ready for vetting by the CodeMeta project for inclusion on that site.

Metadata Bingo

This activity used information from the crosswalking activity and immediately followed it. Its goal was to learn what fields *all* registries and repositories in attendance had in common. After some discussion as to how to proceed, we used SBML, as that resource has a small list of fields, to start matching to the others. This was done much like bingo, in which a field would be called out, and if necessary, described, except that one would call out if one's resource did *not* include that field. We fairly quickly determined that there were only four fields that all of the registries in attendance have in common: name, description, keywords, and location URL.

Creating software citation files for software authors' use

This brief discussion followed the two activities listed directly above, and introduced more fully the idea of creating codemeta.json and CITATION.cff files from existing information, such as that stored by the assembled registries and repositories. It also introduced the next activity; one step of being able to write software to programmatically generated code metadata files is to know how to get to that metadata.

Links to download data from various registries

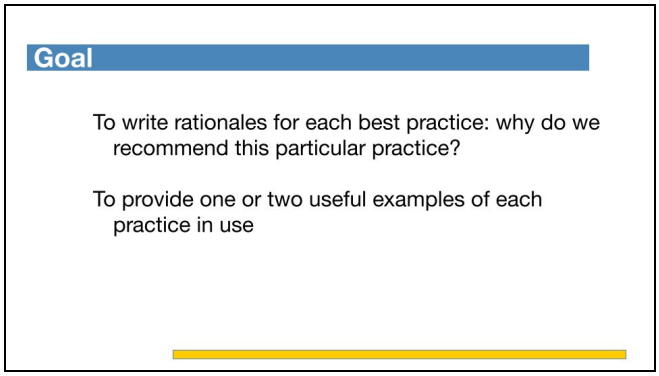
The goal of this short group activity was to create a list in a Google document of the links to download metadata from the various registries that offer download access to this information. Knowing what the metadata in these resources look like is useful for learning more about the individual registries, especially for the workshop's development team, who were tasked with creating tools to help create metadata files to improve citation. This information will also be added to the Repositories Fact Sheet spreadsheet that was created when the initial Task Force was being created. Each registry

representative was asked to supply the link information to the shared Google document. The document has download information for these resources:

ASCL
bio.tools
CaltechDATA
HAL (Software)
OntoSoft
OKG-Soft
SciCrunch
Software Heritage
Zenodo

Breakout groups

The workshop provided time for small group or individual work at three different times over the two days. The first breakout session, on the afternoon of the first day, was focused on the best practices for research software registries that had been identified in earlier conference calls of a task force associated with the FORCE11 Software Citation Implementation Working Group.



The other two breakout sessions were on the second day of the workshop, and the focus of the work done in these later groups was determined in large part by the workshop participants while at the workshop. After each breakout session, representatives from the different groups that had formed reported back to the larger group on the work they had done; discussion often ensued during the report backs, as questions were encouraged and answers could be provided by other members of each breakout group.

Participants were requested to keep notes on their breakout groups in Google docs when possible, and this report links to documents used or created for the breakouts.

Breakout session 1

Best Practices for Research Software Registries breakout group report backs

Repository Scope statement

- We were supportive of the fact that you suggest a repository scope statement should be clearly prominent and easy to find.
- Why do repositories need a scope statement? We have those statements there and elaborated more on the elements.

- This isn't a fully comprehensive list but it's a good start.
- We riffed on a few concepts that - down underneath - questions perhaps others can help us answer, that were outside our scope - and the people who have software repositories provided information on what they have in their scope statement in the examples.

Retention Policy

- By making an entry for an item you're making an implicit promise that it'll stay around. It's important to document that.
- It's important to figure out and list the reasons why information might need to be deleted.
- We might want to say it's a best practice to make persistent identifiers. Users should be able to put in an identifier and have it go somewhere. If the identifier is deleted, the user should get something saying what it was and why it was deleted.
- The retention time of a record fits in the context of an organization as opposed to a fixed amount of time. Resources may allow users to edit or delete records and the platform should be able to capture a history of changes and ensure that what users and administrators are doing is consistent with their policy.

Metadata Schema

- We leveraged a little bit what was already written here, first is that well we should state which schema we are using for representing the data in our repo, right? So this requires or includes the version of the schema you're using, the docs, so people know what you're talking about or expecting.
- Characteristics of expected metadata:
    - Clearly define what entries are required or optional
    - Conventions for all the assumptions in the repo - are the annotations in a certain language? Will some of the values be orcids for authors? These things need to be specified, otherwise you'll have a lot of dirty data.
- Schema should be machine-readable if possible - we should not make that a requirement but it's desirable - if we have this we can automatically validate entries and that's helpful from programmatic pov
- Schema should be crosswalkable with codemeta.
- So we tried to motivate each of these four points with reasons.

Conditions of Use Policy

- This policy should include things like disclaimer, license, and what people can do with the data and metadata in the registry itself, not the codes, but the resource itself.
- The policy lets users of your metadata know how you want to be attributed, a lot of 'to let others know', what your expectations are, what they can and cannot do, whether the metadata can be used commercially, for profit.

- This policy also helps to cover the registry with regard to a legal disclaimer. A registry will likely want to consider disclaiming liability or potential liability for claims that result from misinterpretation or misapplication of the metadata.
- We have some examples but we could not find a good conditions of use policy that covers all of these things. So if you have one, please add it to this document!

End-of-life Policy

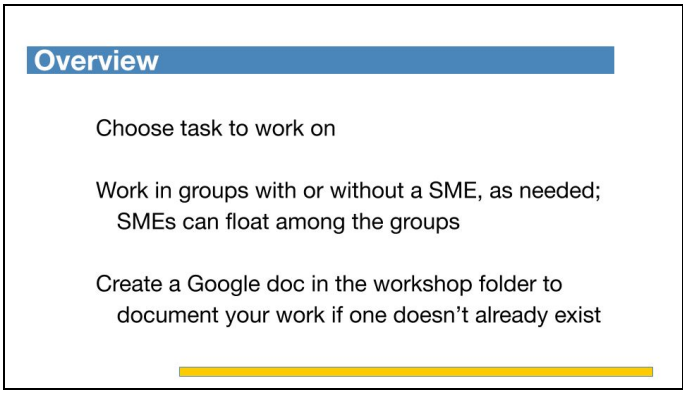
- First we were confused because we thought about the end of life of the entries, the software that we house, and that's all I was interested in.
- We came to the conclusion that each registry entry should be treated with the same dignity at the end of its life as the registry itself. The entry needs to be reproducible, in a repository or something similar so that even if there's no funding and it becomes an orphan, another parent can take it over. We have all these entries that need to go somewhere.

Ethics/Authorship Policy

- There were a lot of things we discussed in terms of authorship.
- Why do this? The rationale, obviously there's credit for people's work which is very important and then having a policy means you can be indexed in SCOPUS usually as a journal. So having the policy puts you at a higher level. If there were levels that we were all aspiring to, we want to be at the next higher level, and at the higher levels you'll have more policies in place.
- You do this because, when you have a dispute, you want a document to point people to. Instead of saying usually we do this or we've never had an authorship dispute before. You don't want to be *that* repo.
- People do fight tooth and nail over journal authorship and if we are theoretically going to be journal-like with these resources we're going to need to deal with a lot more journal-like things such as all of the policies around authorship.
- COPE: Committee on Publication Ethics offers some good examples. The CRediT (Contributor Roles Taxonomy) is not very useful for software.
- One of the things we also discussed which I think this group needs to consider is that an author may not be a human - it may be a consortium of humans and some of those humans are well-hidden by the company that they're working in so they don't have the same kind of authorship.
- Not in complete contrast to journal guidelines, there should be a guideline around the ability to say that code was written by a non-single-human author, an entity.

Breakout session 2

The Thursday morning breakout sessions had one pre-determined group, that for developing codemeta files for software author use from existing metadata; topics



for other groups were suggested by participants and determined after discussion. There were also three people who worked independently, and one person worked with the CodeMeta or CITATION.cff SME separately as needed to work directly on the resource they were improving. Another single-person activity was editing existing best practices documentation.

Breakout session 2 group report backs

- Developing CodeMeta files for software author use
    - People working on a few different approaches for converting local data to codemeta
    - One approach used the crosswalk table built yesterday to map into codemeta. There are some places where that approach doesn't work - with nested objects and dependent logic in certain fields / with certain tags getting mapped, so that we're going to implement separately per-resource(?). Caltech library can implement some of these specialized conversions, so some added issues to keep track of needs as they were found.
    - Some people did manual creation of codemeta files, or doing codemeta built from scratch. That all seemed productive.

- Software value metadata statement
    - We started with the concept of why would anybody in a variety of stakeholders generate metadata to start with? How do we encourage them to do so?
    - We broke down the different groups: researchers that are developers, researchers seeking software, funder community, institution/publications
    - Quickly came to the fact that yes, we all have some skin in the game when it comes to robust metadata.
    - Now we have some value statements that we'll continue to develop and you can all benefit from that.
    - We're really at a difficult time getting consistent use + implementation over >1mil researchers, a lot of researchers to document their software in a consistent way. How can we as repos facilitate this? We're the key here for making sure we have consistent docs.
    - Walking out of this meeting, it'd be great if we had something on our way to an approach. We do want to encourage a level of consistency.
    - We went through workflow from a user's point of view: creating a Github repo, generating your first preserved copy, into zenodo… when do you need to have that metadata file, when does it need to be ready, how can we populate it automatically, what do we need to get from the authors/contributors and at what time and what place, and wouldn't it be great if somehow that live updating repo somehow stored the generated dois or concept-dois, wouldn't it be neat if they can be connected? We didn't quite get that yet.
    - There was another problem that got solved… being able to point to things. If I'm a seeker of a software solution I have discovered some amazing metadata that talks about some amazing computational model from outside my domain and I

don't know quite where that software's located, the DOI and metadata don't necessarily help me. The DOI points to the static version, not the ongoing version, so that was a challenge but the SMEs over here said oh no the cff file / codemeta file could very well have that Github repo or whatever the platform is coming through. So we were really excited about the work that Tom just reported out on that repos are trying to figure out how to populate codemeta/cff file with existing content; that would be helpful.

A good-enough workflow for software citation

- We came up with something that could be termed a good enough and precise enough automated solution to the chicken-egg problem, which means you can keep your citation metadata in your Github repo up-to-date enough for people to appreciate it and work with it. The basic idea is that you have some sort of metadata file in your Github repo - cff or codemeta - and you prepare a release.
- Before a first release it doesn't matter what's in that file; if you have a correct set of authors and the title for example.
- What happens is, you start a release. You tag a commit, push that to Github and it'll be picked up by zenodo, and zenodo will take care of populating the rest of your metadata with up to date things like version string, version DOI, concept DOI and push that to two locations: mainly the actual Github release page. Bin artifacts can be attached after the fact so when people hit this page they'll find an up to date codemeta/cff and will be able to deduce from that how to cite.
- This workflow also accounts for finding a tag that leads not to the release page, but to a part of the repo tree, through the concept DOI.
- We think that's maybe good enough - it'll make things better, people will be able to find specific version info in the metadata.
- We still have some things to work out and write down, which we can do this afternoon.

- Virtual code registry
    - Assuming that all the problems with codemeta and CFF have been resolved, you could think of a set of code repos that you can then talk to from this virtual code registry (VCR) environment which provides the standards and the api that allows other people to build implementations based on that VCR so that one can search across software repos.
    - This could be lightweight (*e.g.,* build a small engine that only grabs author and code name) or it could make a whole rich web environment that does everything and gathers all the information (the structured software metadata) and helps you decipher.
    - We could go inside the package; a package may have 100 different tasks, such as unix or python funcs/classes, that have a name and a small one-line description. So in CFF, I was trying something else - you could have a list of all these tasks and one-line descriptions. If you have that all assembled in the

metadata, that would be useful for scientists to help them do the “reverse search” which is what they called that.

Improving ASCL’s generated CITATION.cff and codemeta.json files

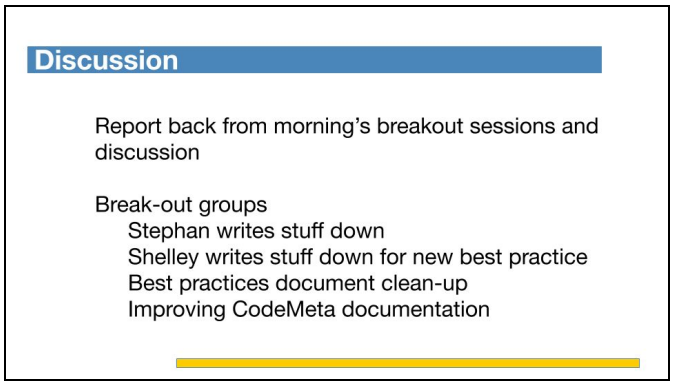
- An ASCL developer worked directly with the CodeMeta and CITATION.cff SMEs to improve the metadata files created from ASCL entries.
- Changes were implemented, and the results were tested and moved in production during the workshop.

Improving CodeMeta

- The website for codemeta that describes all the terms + properties field + definitions for each term imports some ambiguous / not specific enough for software definitions of those terms. It might tell you the class - for code repository you put in a URL but you don’t maybe know exactly what that URL should would look like.
- Feedback from workshop participants on CodeMeta was to have better definitions for all the terms, a little more tight to the crosswalking effort + examples to show what a good value would be + multiple kinds of examples. Different types of software will have more terms with values than other types. So you might have a really full, rich thing for a very well documented repo.
- This needs to start with required elements and then go multiple tiers deeper, so there are multiple depths of description.
- Codemeta probably only has two levels deep; information required for citation and deeper stuff.
- There needs to be more guidance on what I start crosswalking, what do I need to do, what should I try.

Improving the best practices

- For the use policy, I added some questions about what can and cannot be done with metadata with regard to bulk harvesting.
- Though the retention policy is well-developed, I had a question as to whether there should be a minimum time frame in reflection to other standards in your institution, whether it’s seven years for keeping content or whether some institutions might be subject to FOIA rules.



Breakout session 3

All of the work done in this breakout session improved and built upon previous work or documented ideas that had come up during discussions, such as adding a best practice to the existing list. Two groups tackled one issue, that of releasing software, archiving it, and generating good metadata files for citing the software. The report backs for this section made it clear that not only had the individual groups worked well together, but all participants felt comfortable in the group; there was a lot of discussion throughout the report backs, with usually more than one person from a group reporting back and others in the room raising different points and asking questions.

Breakout session 3 group report backs

- Improving CodeMeta documentation

  We talked about the various kinds of improvements needed to the “terms” page. The page was written for one audience and during the crosswalk activity yesterday, it was clear we needed a different page to address a different audience. We need a page with a little more structure, a little more guidance. And there are a couple of other issues with some of the semantics; we opened issues on the CodeMeta Github repo for them. This will allow this discussion to continue there, and document what gets done. We can also open a discussion with schema.org because they have exactly the same issue open, so maybe it can be resolved there. If not, we can add an extension to CodeMeta and that would solve the problems. The end result will be that the next time someone tries to set up one of these mappings at a workshop, it’ll be very transparent as to what these fields mean.

- CFF/CodeMeta/Zenodo matchup: A workflow (“Stephan writes stuff down”)

  This built on and documented an activity in the previous breakout session, “A good-enough workflow for software citation.” I’ve tried to come up with a narrative-ish thing in which a developer does the thing we’ve been talking about. And I hope I’ve included all the details and I hope it’s fairly accessible to the lay-ish reader. So if you can, please read through it and post comments.

- Best practice for users of software registries (“Shelley writes stuff down”)

  We focused on documenting a workflow for software using GitHub and Zenodo. This was incredibly educational. Jurriaan introduced us to the text that he has on his site so we cribbed it, took that copy and then we reorganized it a little bit to make it clear and then he walked us through the process, how you fake out zenodo to make sure you’re using the right authors and title to generate the citation you want. Guess what? This is difficult! And not something you want the generic author to do. A new version of the software to create the initial cff file will

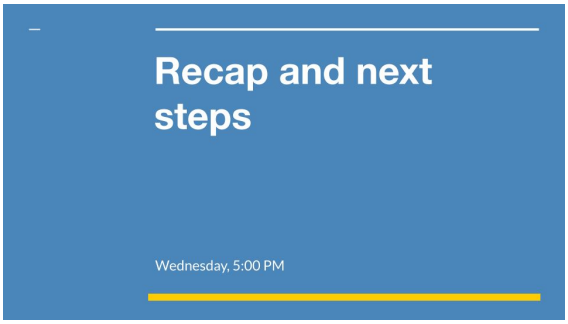
be released soon; it will create 2 files. The software can be embedded in the root of a Github repo, and will be able to do some quality checking of the metadata and generate an accurate citation.

- Best practices documentation clean-up

  We aggregated all of the policies into one document and added a short but clear description. We tried to condense and standardize language, though more work remains to be done. We recommend adding a few things, including examples of policies from your own registries. The Privacy policy still needs work, and there are a few things that also need addtional development. We organized the policy examples thematically at the end of the document. all the way at the bottom I listed what we've done and what needs to be done. Names for credit also need to be added.

Recap and next steps

Sloan funded the workshop and also provided funding for related development work. Before the workshop ended, participants were asked as a group to think about what software development could be done in the next month to advance the work that had been done in the workshop. We used the issues tracker on the workshop's GitHub repo to create action items that everyone could see, comment on, add additional information and ideas, and claim the ones they would like to work on.



Participants representing registries and repositories expressed a desire to continue being in touch with one another, and expanding the group to other registries, such as MAP, SimTK, and swMATH, who've expressed interest in being part of such a group, and one of the issues logged was about setting up this larger consortium. There was a lot of discussion as to what this consortium could do and how it would help those participating in it to help move our respective disciplines to better practices around scientific software. We recognized, however, that first, we needed to finish the work the initial Task Force had set out to do: complete the list of best practices and start to disseminate it.

Ten issues were created to guide our future work, among them ones focused on investigating an initial creation of codemeta.json and citation.cff files, adding our own ORCIDs to the contributor list so we would be able to include these when publishing the list of best practices, codemeta mapping for bio.tools, and creating a GitHub organization for continued work.

Discussion then turned to administration details, such as filing for reimbursements and filling out an evaluation survey, and an opportunity to present information on software metadata files and/or the best practices at the upcoming AGU meeting.

The group expressed gratitude to the Sloan Foundation for funding the event, the workshop organizers thanked the participants for their work, and the workshop ended. The conversations, however, did not!

Evaluation results

The evaluation results were positive; 21 people, all in-person attendees, responded to the evaluation survey, which was conducted at the end of the workshop using a Google Docs form; responses were anonymous. Every respondent rated the workshop overall as Excellent (the highest rating), and 19 of 21 said the workshop fulfilled their expectations; 2 of 21 said their expectations were partially fulfilled but did not take the opportunity to state what would have helped to fulfill them. Similarly, 19 of 21 respondents said they would consider attending another similar workshop in the future, and 2 of 21 replied that they might consider it.

The respondents deemed the “most effective or useful” activities to be the Presentations on others’ registries/repositories (81%), Networking with others (76%), and the Group discussion on shared challenges and solutions (74%).

The question “Did you find the breakout sessions you participated in useful?”, which had an *long answer text* field in which to reply, evoked responses such as:

- Solved significant problems I had at the start of the workshop!
- More than useful, we really found some solutions and paved the way to implementing those that we haven't immplemented (sic) on the spot
- They were very useful! The fact that we had so many diverse groups of registries and repositories really exposed use-cases I wasn't aware even existed.

In the *long text answer* field allowing respondents to “provide any other comments or feedback you have,” replies included:

- It was great. The group really clicked and was really engaged.
- This was extremely useful and helpful. We made progress on pre idenified (sic) and identified in real time issues that would not have been so identified without the back and forth discussions that this workshop enabled. THANKS to all of the organizers for making this happen.
- Yes, it was instrumental in converting our repository's object model metadata into codemeta (currently in staging, will be deployed soon). Getting immediate feedback on what codemeta terms mean from SMEs was very helpful

Detailed responses to each evaluation question are provided in Appendix A.

Changes participating resources made and anticipate making as a result of the workshop

As of January 2021, several participating resources have incorporated the creation of software metadata files into their services. For example, the CoMSES Net Science Gateway has added CodeMeta support to all public landing pages in its Computational Model Library, and a standalone codemeta.json file is included in every computational model download. OntoSoft has added software that extracts metadata from repository README files to create codemeta.json files, and HAL will soon provide a codemeta.json on the fly for each software item it contains (example: https://hal.inria.fr/hal-01897934v3/codemeta). In an effort to try to increase uptake of these files, since the workshop, the ASCL has sent both CITATION.cff and codemeta.json files to a subset of software authors of new ASCL entries and encouraged authors to edit and place one of these metadata files on their software repos.

In alignment with the Best Practice “Share your metadata schema,” several registries and repositories are now making it clearer how their metadata is structured. bio.tools now includes schema.org markup for all entries (example: https://bio.tools/jass), and CoMSES exposes each codebase's metadata as structured metadata on the landing page itself.

DOE OSTI is examining its website to see whether it is in compliance with the best practices the group identified, and CIG is currently revamping its website and will be using the results of this workshop to make sure its repository aligns with best practices. Bio.tools has found a few gaps, such as not having an no end-of-life policy, and will work towards filling these gaps, and ASCL has made bringing its resource into compliance with the best practices a priority for 2021.

Overall, participants expressed interest in continuing to meet as a group; this interest is shared by entities that were not part of the initial task force nor workshop, such as swMATH, SimTK, and MAP. To that end, the workshop organizers have set up a GitHub repo (https://github.com/SciCodes/) and are repurposing an existing domain (https://scicodes.net) for this new consortium of scientific software registries and repositories. We expect the consortium will hold monthly conference calls, and that initially, areas of activity for this new group will include greater implementation of best practices and more work around software metadata files, both creating them and working to increase their adoption in our respective disciplines.

Lessons learned about planning and executing the workshop

Overall, both the planning for and the workshop itself progressed smoothly. The team worked well together, met virtually as needed, and employed project management strategies to keep the workshop implementation on track. Most participants had met and worked together before; those who were new to the group participated eagerly and fully, and appeared comfortable even without prior experience with the others. We had coffee and food available before the workshop

started and during breaks, which allowed people to mingle freely. We also had various activities in the plenary time and in breakout sessions that encouraged people to work with others in various configurations.

The balance of participants was appropriate; several people expressed that we “had the right people in the room” to make the meeting productive and useful. Though the subject matter experts had been invited to share their expertise with the registry/repository managers, all of them said they got a lot out of attending, learned more about how to better work with our resources, and were glad they were included.

We might have done well to have a few different kinds of users available, perhaps remotely at specific times, to provide input and feedback; this could include software developers, journal data editors, and researchers. Though some participants fit into the *developer* and/or *researcher* role, we might have benefitted by getting those with less experience with software metadata, citation issues, and registry management involved.

P. Wesley Ryan, a part-time volunteer developer for the Astrophysics Source Code Library, served as scribe. He took detailed notes throughout the two days during plenary sessions and discussions. This was very useful and we feel that future workshops would do well to request funding to cover the cost of a scribe who is knowledgeable with the vocabulary of the participants (and types really really fast).

As some participants had not been involved in creating the drafts of the Best Practices documents, Alice Allen should have gone through these documents more thoroughly with the assembled group before starting the breakout session for them. Even better might have been to send these out ahead of time so people could read them, and then also go through them before the breakout sessions. It took a little time in the Best Practices breakout sessions to explain the documents and how they had been developed to those new to them.

Our implementation of remote service through Webex could have been more effective. We offered it at the request of a couple of people who couldn’t attend yet wanted to participate; remote participation had not been part of the initial planning for the event. If we offer Webex/remote service again, we should have someone dedicated to managing the remote experience. For breakout activities, we could offer one breakout group that is *only* online (all participants are remote). A breakout group that is a hybrid of in-person and online participants was difficult to manage effectively; a pairs activity that is one in-person and one online participant could work well, though.

We should have allocated specific time for questions after each presentation. Though presenters knew how much time had been allocated to their talks, it was not clear to them that if they wanted to take questions, they needed to carve out time to do that within their allocation. Alternately, we could have built two or three minutes for Q&A for each presentation into the schedule over and above what we told them was their allotted time.

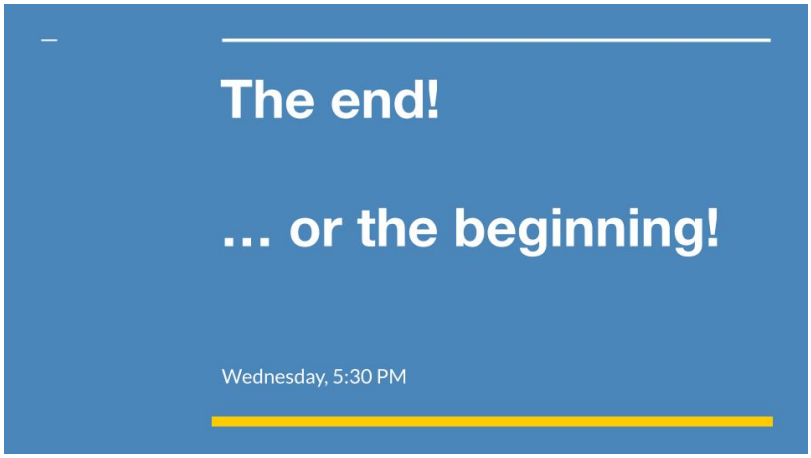
Getting the abstracts for the presentations before the workshop would have been helpful. We should also have required the slides the day before the presentation to enable us to test them beforehand, as that would have avoided an incompatibility issue we ran into.

We could have easily used more time, and if we hold a future workshop, should consider making it longer and using some of the time to go more deeply into how the different registries work and how they arrived as some of their decisions to function as they do. Though adding time for Q&A at the end of the registry presentations would have helped meet the participants' desire to know more, offering more open and unscripted discussion time could be even more useful.

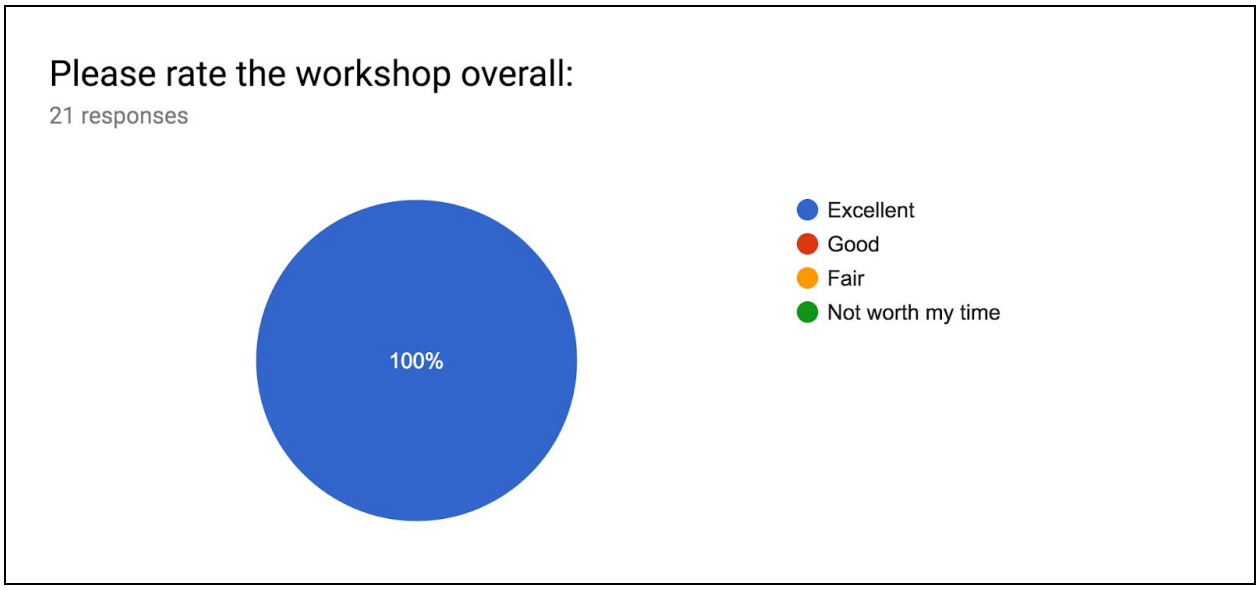
Future action

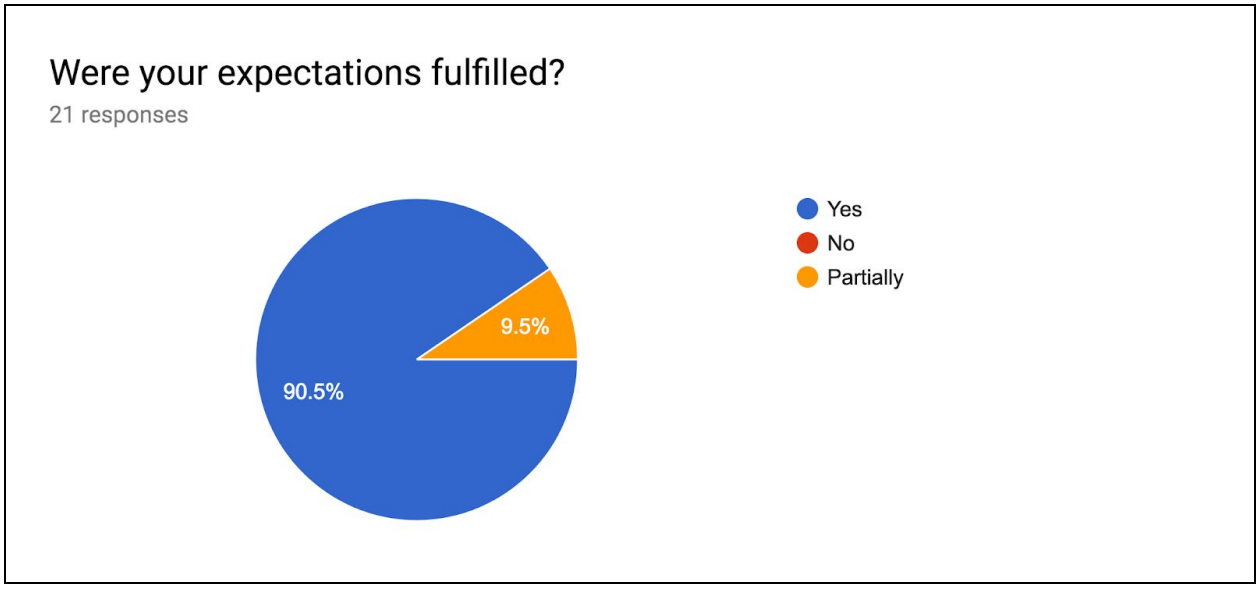
With many involved in the Task Force and workshop expressing a desire to continue to meet and expand the group to other registries that have expressed interest, such as MAP and SimTK, the workshop organizers have established a GitHub site and domain (scicodes.net) for this effort. A coalition, currently referred to as SciCodes, would be:

- For editors and maintainers of academic discipline and institutional software registries and repositories and those who work with these resources
- To share work methods, marketing ideas, and communication practices
- To demonstrate unique aspects of our respective services, discuss challenges and share solutions to common issues that arise in managing our resources
- To work cooperatively to speed adoption of the CodeMeta and CFF standards and better enable software citation, recognition, and dissemination
- To work toward a virtual registry standard to enable searching across multiple registries



Appendix A: Evaluation survey responses





If your expectations were not met, please let us know what would have helped to meet them:

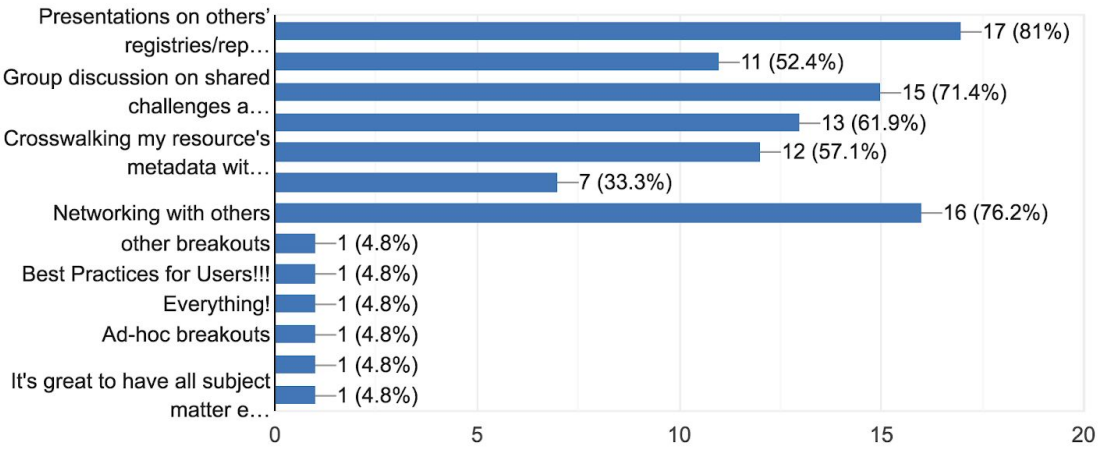
2 responses

How do we do this again?

Actually beyond my expectation :)

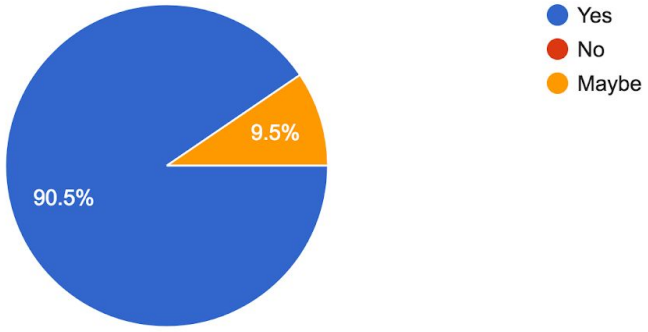
What part(s) of the workshop were most effective or useful for you?

21 responses



Would you consider attending another Registry Collaboration Workshop in the future, should another be held?

21 responses



What breakout sessions did you participate in (other than the Best Practices breakout)?

19 responses

a bunch

multiple

Best Practices for Users

Developing CodeMeta files for software author use, Best practices documentation clean-up (oh ... gosh ...)

Ethics

Devising a good enough workflow for software citation, "Stephan writes stuff down" (i.e., documenting the good enough workflow, Implementing CFF generation from ASCL entries

Schema best practice; Codemeta documentation

Best practices

codemeta conversion, automation

Metadata schema / Crosswalk to CodeMeta

CodeMeta, Conditions of Use Policy

CodeMeta doc improvement, CITATION.cff v1.1.0 implementation

improve codemeta terms documentation, codemeta crosswalk

1. The breakout session for "a good enough workflow for software citation". We explored ways to automate the process for incentivizing the real implementation of software citation at each software creator's behavioral level. 2. The crosswalk session. I was with Software Heritage and biotools doing the crosswalk.

CodeMeta/CFF/Zenodo workflow

CalTech crosswalk & Best Practices 2.0

registry scope; zenodo cff integration; best practices for users

Thursday breakout session on communicating the value of metadata to different stakeholders

Did you find the breakout sessions you participated in useful? Please elaborate on your answer if you wish; we would like to know how the session(s) were or will be useful.

18 responses

yes!

yes

Solved significant problems I had at the start of the workshop!

yes, everything was very useful, and gave me food for thoughts: exchanging with "registry owner" peers will help me improve my resource(s) and collaborate with some of them

yes. back and forth discussions and the tangents were very helpful in illuminating common issues and possible solutions.

Yes, because they were smaller and we could dive pretty deep on things.

More than useful, we really found some solutions and paved the way to implementing those that we haven't immplemented on the spot

Yes, the one for schema best practices was extremely useful to flesh out why it should be recommended as a best practice. The second one (codemeta documentation) was more emphasized on highlighting issues with the current page than fixing them, but I hope it is a nice first step towards improving the schema.

The breakout sessions were useful, but getting folks into groups was a bit painful. Having more separate breakout spaces might have been better.

useful to share point of view, and useful for well understand codemeta cff and the lack in our repositories

Yes, not only for direct examination and action on their central topics but also for deep consideration of related issues.

Yes - answers were clarifying as an implementer and I think the dev feedback they get will be taken into account in future docs.

Yes, it was instrumental in converting our repository's object model metadata into codemeta (currently in staging, will be deployed soon). Getting immediate feedback on what codemeta terms mean from SMEs was very helpful

Yes, it is especially helpful for CiteAs. First, as a technical participant in the software citation toolchain, it's very important for us to understand how the software metadata can run through the key stakeholders in the scientific software ecosystem, so we can get real insights about how we can better work with this techical workflow. Second, for the Best Practices breakouts, it's very important for us to understand the progress, challenges, and concerns that software registries/repositories are faced with. So we can better understand the needs of software registries/repositories and seek better ways to serve this community. It greatly helps bridge the conversation between registries/repositories and subject matter experts too. Third, I also have had very meaningful conversation with other subject matter experts so we will be able to brainstorm better approach to improve software citation. If this workshop did not take place, we would not be able to have this conversation, exchange experiences and seed promising collaborative practices and solutions.

They were very useful! The fact that we had so many diverse groups of registries and repositories really exposed use-cases I wasn't aware even existed.
Yes. It was very helpful to see others create their crosswalks & the followup discussions were additionally useful.
scoping was useful because its a problem im currently experiencing with my resource; zenodo cff integration was useful for networking and practical exploration of what could be done; best practices for users was useful but more of a push from my end since we followed the escience best practice.
Yes! The breakout was actually mostly diverted toward the topic of understanding technical implementations of metadata and DOIs in various major repos/dev platforms. But this was useful, too.

Will you be implementing anything covered or learned in this workshop in your future work? If so, what?

19 responses

yes!

specific tasks were very good to have

Absolutely! Guiding researchers to domain repositories, techniques for accurate citation for metadata, tools for improving the connection between GitHub and Zenodo.

Yes! We got a lot of help with improving how we generate CodeMeta and CFF files for users and have already implemented some of the necessary changes, and will be implementing the rest shortly.

yes, hopefully, compatibility with codemeta/cff

Yes. codemeta and other pieces of software that I learned about that will improve what I do!

Yes. I'l be bringing a lot of feedback back to the codemeta group

The good enough workflow!

I have implemented the mapping from OKG-Soft to Codemeta. I plan to release the OntoSoft-Codemeta as well soon. I am working on pre-populating Codemeta files automatically from GitHub, I hope to have some results soon

I will be adding citation.cff and codemeta.json support in the (new) sbml.org software guide

I look forward to implementing a lot of the ideas picked up at the workshop

i will propose to iimplement codemata on cff on fly for each item in repositorie

Implementing practices in my own repository, refining its crosswalk, and following up on action items related to best practices.

Yes - already implemented some CITATION.cff updates, will continue working on improving our CodeMeta/CITATION.cff stuff.

Finally integrating codemeta into our repository

Yes. First, CiteAs will seek to be integrated into GitHub so our current technical capabilities can be leveraged by software repositories on GitHub to check the citability of their code projects. Second, we had some conversation with software repositories and we will be able to change our curated data about software mentions in scientific publications to build better machine learning system to incentivize researchers to take on better software citation practice. We can do more after digesting what we've gained from the workshop. The networking opportunities also help us a great lot. Looking forward to do more to serve this community.

CodeMeta/CFF integration (importing/exporting), and eventually to also be able to include in our documentation exceprts from the best practices and also point back to them.

Yes, especially regarding Best Practices. We will be using these to build our of polcies and implement our INVENIO repository.

already have cff and codemeta for every version of all software in my resource; i'll work on implementing a new download a .zenodo.json button for cffinit web site

How could the workshop have been improved?

14 responses

a little more time for questions after presentations

was a tad of codemeta dogma

perfect as is ;)

More time - well not really, to work one-on-one with codemeta experts.

I really thought a 10 min presentation on codemeta would have been sufficient prior to arriving but realized, especially by day 2, that I could've used a ~20-30 minute slot.

n/a

More time to discuss future steps would have helped. Other than that my only problem is that the time went by too fast :)

I often wanted to ask questions of the people presenting their work, but we didn't build enough time for that into the schedule.

it was good i don't know it could be improved!

So far everything is great.

I think introducing official parts/timeslots for Hackathons (for rapid code prototyping) and "Writeathons" (for producing documentation and website content) would multiply the outcomes

Perhaps a code of conduct to be discussed at the begining of the workshop (c.f. https://datacurationnetwork.org/about/code-of-conduct/)

it was great! minor thing for me was the crosswalking because it was not so useful for me, and i see some fundamental issues with crosswalking, e.g. https://github.com/citation-file-format/citation-file-format/pull/77#issuecomment-550250708

I only wish there were more opportunities for topic-switching in breakouts. There were too many different groups I wanted to participate in! I am not sure how this would be feasible, though.

Please provide any other comments or feedback you have.

10 responses

good organization! everything worked well!

Having the right people in the room made all the difference for this workshop! Thank you to Alice, Tom and our funder Sloan!

This was extremely useful and helpful. We made progress on pre identifed and identified in real time issues that would not have been so identified without the back and forth discussions that this workshop enabled. THANKS to all of the organizers for making this happen.

It was great. The group really clicked and was really engaged.

Not sure if this makes sense, but the elephant in the room (sometimes) seemed to have been GitHub and other source code repo platforms. If we can identify individuals that would be interested in a collaboration, it may be helpful to invite them for part or all of another workshop. Or maybe that's another workshop/discussion to be had.

It was wonderful and Alice did an exceptional job.

"Bravo" for the organisation !

very well organized and structured with valuable group of participants

We would love to serve as, and act with, subject matter experts to seek better ways to serve this community.

We had representation from registries, repositories, journals and funders (or at least closely related to funders), but not from the end-users, i.e. developers (RSEs) and paper authors (that cite software). Their input (if not through the entire workshop, at least at some key points), would really help with assessing how pragmatic our outcomes are.