Making Correct Software Citation Easy and Useful

Aims (common minimum goals)

A. Define what correct software citation is.

1. Begin with FORCE11 guidelines
    a. Cite version-specific DOIs for major versions when using the software in the work.
    b. Cite general DOIs for citations of the concepts in the software (e.g. concept DOIs on Zenodo).
    c. Cite reference publications associated with software (e.g. JOSS publications) in both cases.
2. What changes to these guidelines are needed? Why?
    a. E.g., what about Software Hash IDs? You can extract a citation from SWH like you can for GitHub. This citation will contain DOI if available and SWHID.https://www.softwareheritage.org/2025/05/07/software-heritage-citation-feature/
    b. Citation is about credit, identification (the reference) is not enough for giving proper credit- this is relevant for standardization especially when there are many authors involved
3. Are there additional portions of this that can be standardized (e.g. [Software] vs [Software Application] vs [SoftwareApp] in the citation in the publication)?

B. What are the relevant use cases for software citation?

1. Proper credit attribution
    a. RSEs and similar need proper attribution to aid in retention and performance reviews. This makes science more efficient by spending less time/$ training new talent.
2. Research transparency and validation
    a. Version-specific software citations critical to understanding the published result
    b. Version-specific software supports reproducibility of research results
3. To support assessments of software impact (e.g. to determine whether it should receive funding)
    a. Number of times a software package is cited in peer-reviewed publications.
    b. Number of times a software package is cited in other items, such as data, software environments, etc.
4. Policy compliance and performance assessments
    a. E.g. NASA is beginning to show signs of assessing research publications by how many cite data and software. Also showing a push to require ORCiDs in those publications' metadata. Likely to extend this to data and software metadata.
    b. An institution wants to discover what software its employees have created or are contributing to. It searches for software artifacts with the institution's ROR in the affiliation metadata of various platforms, but fails to find a significant number of

software packages, despite having prior knowledge of a large number of software efforts.

5. Generation of Complex Citation Objects (e.g. for containerized software environments with numerous software packages).
    a. Must be able to programmatically extract version-specific software citations for tens or hundreds of software packages to properly cite them when creating containerized software environments.
6. Person wants to acknowledge various contributors to a given software package and uses software to help them include that recognition in their metadata using a controlled taxonomy (e.g. allcontributors.org).
    a. However, those role names are not compatible with those in DataCite. As a result, the contribution is not properly acknowledged (if at all) in the DataCite metadata, creating an incorrect citation.
    b. Today there are multiple places that contributors (and roles) can be stored (e.g., allcontributors, CITATION.cff, codemeta.json, zenodo.json, journal article author lists (and CRediT roles), etc.) and there aren't mechanisms to make/keep them consistent

C. Plan how to make correct software citation easy.

1. What infrastructure changes are needed for software citations to be correct? Why?
    a. ~~Include “Cite As” in DataCite and Schema.org to indicate the correct citation to software (e.g. when funders require a certain accompanying statement). Why? - Need to indicate this in both to ensure people can access the correct citation in the DOI services (e.g. citation.doi.org) and via landing pages.~~ *Discussion: Doing this would imply a major change to be required of the CrossCite citation service to supply the given metadata in the returned citation. Until a case driven by demand demonstrated with high statistics is presented, this is out of scope.*
    b. Include “Cite With” in DataCite and Schema.org to indicate what items should be cited with the software (e.g. reference publications). Why? - Need to indicate this in both to ensure people can easily find what items to cite with the software in the DOI services (e.g. citation.doi.org) and via landing pages.
    c. Improve mapping from DataCite to Schema.org so the output validates in Schema.org in a codemeta-compliant approach (e.g. mapping for contributor roles from DataCite does not validate). Why? - To improve the alignment between Schema.org embedded on landing pages with the DataCite metadata and CodeMeta terms. The Schema.org json-ld on the landing page will be ignored if there are critical issues, which is currently the case for the DataCite Schema.org json-ld output.
    d. Allow use of taxonomies for contributor roles in DataCite metadata (e.g. allcontributors.org). Why? - Necessary to enable proper recognition of contributions to software. *Discussion: Support for this requires a breaking change, to be discussed in an upcoming major release. Idea for a minor fix: https://github.com/datacite/datacite-suggestions/discussions/32*

    e. Alignment of citation information across platforms and metadata files compared to DataCite, FORCE11, and any updates needed. Why? - Publishers consider DataCite to be the “source of truth” for citation information for data and software. All citations provided need to align with that, and also align with standards / best practices for correct software citation. This prevents wrong citations from being used.
    f. Fix the pipeline from GitHub to DataCite for correct BibTex citation generation (citation.doi.org). Currently indicating software as books.
2. What infrastructure changes are needed to make software citations easily accessible? Why?
    a. Embed software citation mechanisms into Python and other major languages.
    b. Reduce the complexity of the metadata file landscape for software by first aligning the information across these files and later converging into a single file to serve as the intersection point. Why? - software developers will rarely spend time creating metadata files, especially the growing number being demanded of them. In the end, we need one correct file that is designed to serve multiple purposes, not one file per purpose. (See slide deck for relevant stakeholders on this.)
    c. Offer correct citation information through software installation and management services (e.g. pip/conda for Python). One pathway is to add a command to each to get the citations for all the packages installed in a particular environment on one’s computer.
3. What infrastructure changes are needed to make software citation metadata more easily useful to various stakeholders?
    a. GitHub/similar APIs need to provide ORCiDs, affiliation names, and affiliation ROR information in their API services. Why? - assessment of policy compliance, tracking of software produced by a person or institution/entity.
    b. Add funding information, including funder RORs, to those API services. Why? - Track software produced by a given funder to assess impact.
    c. Add software DOIs to those API services. Why? - Funders and institutions will want to track how many times and how often a software is used by the community, so they
        - (a) will pull the DOI from these APIs, pull the concept DOI and those indicated in the “Cite As” and “Cite With” fields in DataCite, and then use available citation tracking software to retrieve citation statistics on those DOIs.
        - (b) pull the DOI from DataCite or documentation internal to that funder or institution and query the hosting repository’s API for more detailed statistics associated with that DOI, such as number of downloads, posted/resolved issues, and similar statistics for assessing community involvement in and use of that software.
    d. Add support for use of controlled vocabularies in the DataCite Subject field on generalist archival platforms (e.g. Zenodo) that are external to OpenAire and DataCite. Why? - Science-specific software registries can then search for

software packages using specific terms from relevant vocabularies for potential additional curation efforts.

Potential case studies/pilots
NASA or specific disciplines such as heliophysics or earth sciences
MOSS

Collaboration between

1. ReSA's Research Software Infrastructure Forum (RSI) Forum focus on software citation use case
2. Jonathan Starr - led Impact of Research Software Open Innovation Sprint
3. Daniel Garijo of CodeMeta
4. NASA's closing the gaps in software citation initiative - SciCodes meeting presentation: slides https://doi.org/10.5281/zenodo.15237081 and the link to the recorded presentation https://youtu.be/as6KpOGR8bE. Slide deck link: ImprovingSoftwareCitation (see last three slides for differences)

Organisations representatives/Individuals already involved:
Daniel S. Katz (NCSA, ReSA), Jonathan Starr (NumFOCUS and other things), Michelle Barker, Nicola Tarocco (Zenodo, DataCite Metadata working group, RSI Forum), Guido Juckeland (HZDR, Helmholtz and EVERSE), Daniel Garijo Verdejo (SciCodes, CodeMeta, Impact of Research Software Open Innovation Sprint), Marcel Meistring (Helmholtz Open Science Office), Kaylin Bugbee (NASA/DASP), Alex Ioannidis (CERN/Zenodo/InvenioRDM), Kristina Vrouwenvelder (AGU), Brian Thomas (NASA/HDRL), Rebecca Ringuette (NASA), Anna Kelbert (ADS/SciX), Shelley Stall (AGU)

Other organisations needed:
GitHub
Languages eg Python?
DataCite
Other publishers
Peer review communities (e.g., rOpenSci, PyOpenSci)
Software installation services (e.g. conda, pip, etc for Python)
Package managers
Schema.org

Organisational structure
Could be a working group of the RSI Forum, co-convened with other organisations/initiatives, to utilise organisational identities to add weight

Process

1. Determine potential activities

2. For each, determine who needs to be involved for it to be successful, and about how much time will be needed to complete it, and if it is dependent on any other activities or external factors, relative importance/impact, and chance of successful completion
3. Prioritize, based on factors above along with willingness of someone to lead and willingness of others to be involved