Introduction to Software Citation Principles

Daniel S. Katz
Assistant Director for Scientific Software & Applications, NCSA
Research Associate Professor, CS
Research Associate Professor, ECE
Research Associate Professor, iSchool
dskatz@illinois.edu, d.katz@ieee.org, @danielskatz

FORCE11 Scholarly Communications Institute
WT02: Software Citation: Principles, Usage, Benefits, and Challenges
2–3 August 2017
Software citation principles: People & Process

- FORCE11 Software Citation group started July 2015
- WSSSPE3 Credit & Citation working group joined September 2015
- ~55 members (researchers, developers, publishers, repositories, librarians)
- Reviewed existing community practices & developed use cases
- Drafted software citation principles document
  - Started with data citation principles, updated based on software use cases and related work, updated based working group discussions, community feedback and review of draft, workshop at FORCE2016 in April
  - Discussion via GitHub issues, changes tracked
- Submitted, reviewed and modified (many times), now published
Software citation principles paper

- Contents (details on next slides):
  - 6 principles: Importance, Credit and Attribution, Unique Identification, Persistence, Accessibility, Specificity
  - Motivation, summary of use cases, related work, and discussion (including recommendations)
- Format: working document in GitHub, linked from FORCE11 SCWG page, discussion has been via GitHub issues, changes have been tracked
- Reviews and responses also in PeerJ CS paper
Principle 1. Importance

• Software should be considered a legitimate and citable product of research. Software citations should be accorded the same importance in the scholarly record as citations of other research products, such as publications and data; they should be included in the metadata of the citing work, for example in the reference list of a journal article, and should not be omitted or separated. Software should be cited on the same basis as any other research product such as a paper or a book, that is, authors should cite the appropriate set of software products just as they cite the appropriate set of papers.
Principle 2. Credit and Attribution

- Software citations should facilitate giving scholarly credit and normative, legal attribution to all contributors to the software, recognizing that a single style or mechanism of attribution may not be applicable to all software.
Principle 3. Unique Identification

- A software citation should include a method for identification that is machine actionable, globally unique, interoperable, and recognized by at least a community of the corresponding domain experts, and preferably by general public researchers.
Principle 4. Persistence

- Unique identifiers and metadata describing the software and its disposition should persist – even beyond the lifespan of the software they describe.
Principle 5. Accessibility

- Software citations should facilitate access to the software itself and to its associated metadata, documentation, data, and other materials necessary for both humans and machines to make informed use of the referenced software.
Principle 6. Specificity

• Software citations should facilitate identification of, and access to, the specific version of software that was used. Software identification should be as specific as necessary, such as using version numbers, revision numbers, or variants such as platforms.
### Use cases

**Table 2.** Use cases and basic metadata requirements for software citation, adapted from [20]. Solid circles (•) indicate that the use case depends on that metadata, while plus signs (+) indicate that the use case would benefit from that metadata if available.

<table>
<thead>
<tr>
<th>Use case</th>
<th>Unique identifier</th>
<th>Software name</th>
<th>Author(s)</th>
<th>Contributor role</th>
<th>Version number</th>
<th>Release date</th>
<th>Location/repository</th>
<th>Indexed citations</th>
<th>Software license</th>
<th>Description</th>
<th>Keywords</th>
<th>Example stakeholder(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Use software for a paper</td>
<td>• • •</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>+</td>
<td>+</td>
<td></td>
<td>+</td>
<td>•</td>
<td>•</td>
<td>Researcher</td>
</tr>
<tr>
<td>2. Use software in/with new software</td>
<td>• • •</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>+</td>
<td>•</td>
<td></td>
<td>+</td>
<td>•</td>
<td>•</td>
<td>Researcher, software engineer</td>
</tr>
<tr>
<td>3. Contribute to software</td>
<td>• • • •</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>+ •</td>
<td>•</td>
<td></td>
<td>+</td>
<td>•</td>
<td>•</td>
<td>Researcher, software engineer</td>
</tr>
<tr>
<td>4. Determine use/citations of software</td>
<td>• • • •</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>+ •</td>
<td>+</td>
<td></td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>Researcher, software engineer</td>
</tr>
<tr>
<td>5. Get credit for software development</td>
<td>• • • +</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>+ •</td>
<td>+ •</td>
<td>+</td>
<td></td>
<td>+</td>
<td>•</td>
<td>•</td>
<td>Researcher, software engineer</td>
</tr>
<tr>
<td>6. “Reproduce” analysis</td>
<td>• • • • •</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>+ •</td>
<td>+</td>
<td></td>
<td>+</td>
<td>•</td>
<td>•</td>
<td>Researcher</td>
</tr>
<tr>
<td>7. Find software to implement task</td>
<td>• • • • •</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>+ • •</td>
<td>+</td>
<td></td>
<td>+</td>
<td>•</td>
<td>•</td>
<td>Researcher, software engineer</td>
</tr>
<tr>
<td>8. Publish software paper</td>
<td>• • • • •</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>+ • • •</td>
<td>+</td>
<td></td>
<td>+</td>
<td>•</td>
<td>•</td>
<td>Publisher</td>
</tr>
<tr>
<td>9. Publish papers that cite software</td>
<td>• • • • • •</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>+ • • •</td>
<td>+</td>
<td></td>
<td>+</td>
<td>•</td>
<td>•</td>
<td>Publisher</td>
</tr>
<tr>
<td>10. Build catalog of software</td>
<td>• • • • •</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>+ •</td>
<td>•</td>
<td></td>
<td>+</td>
<td>•</td>
<td>•</td>
<td>Indexer</td>
</tr>
<tr>
<td>11. Build software catalog/catalog registry</td>
<td>• • • • • •</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>+ •</td>
<td>•</td>
<td></td>
<td>+</td>
<td>•</td>
<td>•</td>
<td>Domain group, library, archive</td>
</tr>
<tr>
<td>12. Show scientific impact of holdings</td>
<td>• • • • • •</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>• •</td>
<td>+</td>
<td></td>
<td>+</td>
<td>•</td>
<td>•</td>
<td>Repository</td>
</tr>
<tr>
<td>13. Show how funded software has been used</td>
<td>• • • • • • •</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>• •</td>
<td>+</td>
<td></td>
<td>+</td>
<td>•</td>
<td>•</td>
<td>Funder, policy maker</td>
</tr>
<tr>
<td>14. Evaluate contributions of researcher</td>
<td>• • • • • • • •</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>• • •</td>
<td>+</td>
<td></td>
<td>+</td>
<td>•</td>
<td>•</td>
<td>Evaluator, funder</td>
</tr>
<tr>
<td>15. Store software entry</td>
<td>• • • • • • • •</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>• • •</td>
<td>•</td>
<td></td>
<td>+</td>
<td>•</td>
<td>•</td>
<td>Citation manager</td>
</tr>
<tr>
<td>16. Publish mixed data/software packages</td>
<td>• • • • • • • •</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>• • •</td>
<td>•</td>
<td></td>
<td>+</td>
<td>•</td>
<td>•</td>
<td>Repository, library, archive</td>
</tr>
</tbody>
</table>

[20] FORCE11 Software Citation Working Group. Software citation use cases. https://docs.google.com/document/d/1-ldS0SqGoBIFwLB5G3HiLLEOSAAgMdo8QPEpjYUaWCv1U
Related work

• General community
  • Blogs & papers studying the issue by groups (e.g., SSI), people (e.g., Wilson), and workshop reports (e.g., by WSSSPE and SSI)

• Domain-specific
  • Work by journals to encourage software publication & citation (e.g., TOMS, AAS, ASCL, NIH SDI, Ontosoft)

• Metadata-focused
  • For citation: DOAP, Research Objects, The Software Ontology, EDAM Ontology, Project CRediT, Ontosoft, RRR/JISC guidelines
  • Also for build/distribution: Debian package format, Python package descriptions, R package descriptions
  • CodeMeta crosswalk activity to be discussed
Discussion: What to cite

• Importance principle: “…authors should cite the appropriate set of software products just as they cite the appropriate set of papers”
• What software to cite decided by author(s) of product, in context of community norms and practices

• POWL: “Do not cite standard office software (e.g. Word, Excel) or programming languages. Provide references only for specialized software.”
• i.e., if using different software could produce different data or results, then the software used should be cited
Discussion: What to cite (citation vs provenance & reproducibility)

• Provenance/reproducibility requirements > citation requirements
• Citation: software important to research outcome
• Provenance: all steps (including software) in research
• For data research product, provenance data includes all cited software, not vice versa
• Software citation principles cover minimal needs for software citation for software identification
• Provenance & reproducibility may need more metadata
Discussion: Software papers

- Goal: Software should be cited
- Practice: Papers about software (“software papers”) are published and cited
- Importance principle (1) and other discussion: The software itself should be cited on the same basis as any other research product; authors should cite the appropriate set of software products
- Ok to cite software paper too, if it contains results (performance, validation, etc.) that are important to the work
- If the software authors ask users to cite software paper, can do so, in addition to citing to the software
Discussion: Derived software

- Imagine Code A is derived from Code B, and a paper uses and cites Code A
- Should the paper also cite Code B?
- No, any research builds on other research
- Each research product just cites those products that it directly builds on
- Together, this give credit and knowledge chains
- Science historians study these chains
- More automated analyses may also develop, such as transitive credit

Discussion: Software peer review

- Important issue for software in science
- Probably out-of-scope in citation discussion
- Goal of software citation is to identify software that has been used in a scholarly product
  - Whether or not that software has been peer-reviewed is irrelevant
- Possible exception: if peer-review status of software is part of software metadata
- Working group opinion: not part of the minimal metadata needed to identify the software
Discussion: Citations in text

• Each publisher/publication has a style it prefers
  • e.g., AMS, APA, Chicago, MLA
• Examples for software using these styles published by Lipson
• Citations typically sent to publishers as text formatted in that citation style, not as structured metadata
• Recommendation: **text citation styles should support:**
  • a) a label indicating that this is software, e.g. [Computer program]
  • b) support for version information, e.g. Version 1.8.7

Discussion: Citation limits

- Software citation principles
  - more software citations in scholarly products
  - more overall citations
- Some journals have strict limits on
  - Number of citations
  - Number of pages (including references)
- Recommendations to publishers:
  - Add specific instructions regarding software citations to author guidelines to not disincentivize software citation
  - Don’t include references in content counted against page limits
Discussion: Unique identification

• Recommend DOIs for identification of published software
• However, identifier can point to
  1. a specific version of a piece of software
  2. the piece of software (all versions of the software)
  3. the latest version of a piece of software
• One piece of software may have identifiers of all 3 types
• And maybe 1+ software papers, each with identifiers
• Use cases:
  • Cite a specific version
  • Cite the software in general
  • Link multiple releases together, to understanding all citations
Discussion: Unique identification (cont.)

- Principles intended to apply at all levels
- To all identifiers types, e.g., DOIs, RRIDs, ARKS, etc.
- Though again: recommend when possible use DOIs that identify specific versions of source code
- RRIDs developed by the FORCE11 Resource Identification Initiative
  - Discussed for use to identify software packages (not specific versions)
  - FORCE11 Resource Identification Technical Specifications Working Group says “Information resources like software are better suited to the Software Citation WG”
- Currently no consensus on RRIDs for software
Discussion: Types of software

- Principles and discussion generally focus on software as source code.
- But some software is only available as an executable, a container, or a service.
- Principles intended to apply to all these forms of software.
- Implementation of principles will differ by software type.
- When software exists as both source code and another type, cite the source code.
Discussion: Access to software

• Accessibility principle: “software citations should permit and facilitate access to the software itself”
• Metadata should provide access information
• Free software: metadata includes UID that resolves to URL to specific version of software
• Commercial software: metadata provides information on how to access the specific software
  • E.g., company’s product number, URL to buy the software
• If software isn’t available now, it still should be cited along with information about how it was accessed
• Metadata should persist, even when software doesn’t
Discussion: Identifier resolves to …

• Identifier that points directly to software (e.g., GitHub repo) satisfies Unique Identification (3), Accessibility (5), and Specificity (6), but not Persistence (4)
• Recommend that **identifier should resolve to persistent landing page that contains metadata and link to the software itself, rather than directly to source code**
• Ensures longevity of software metadata, even beyond software lifespan
• Point to figshare, Zenodo, etc., not GitHub
Example 1: Make your software citable

• Publish it – if it’s on GitHub, follow steps in https://guides.github.com/activities/citable-code/

• Otherwise, submit it to zenodo or figshare, with appropriate metadata (including authors, title, …, citations of … & software that you use)

• Get a DOI

• Create a CITATION file, update your README, tell people how to cite

• Also, can write a software paper and ask people to cite that (but this is secondary, just since our current system doesn’t work well)
Example 2: Cite someone else’s software in a paper

- Check for a CITATION file or README; if this says how to cite the software itself, do that
- If not, do your best following the principles
  - Try to include all contributors to the software (maybe by just naming the project)
  - Try to include a method for identification that is machine actionable, globally unique, interoperable – perhaps a URL to a release, a company product number
  - If there’s a landing page that includes metadata, point to that, not directly to the software (e.g. the GitHub repo URL)
  - Include specific version/release information
- If there’s a software paper, can cite this too, but not in place of citing the software
Software Citation vs Paper Citation

• Three relevant steps for paper citation
  1. Creator (aka author) submits paper to “publisher”
  2. [review+], then publisher publishes paper & assigns identifier, often DOI
  3. To refer to paper within another work, cite paper metadata, often including DOI
• Fixed order, discrete steps
• For software today
  • Creator develops software on GitHub, released at different stages (versions) during its development
  • Someone who uses that software will likely not cite it, but if they do, they will cite the repository
  • No step 2
  • Partial step 3, because there is no clear metadata or identifier for the software that was used
• Software citation principles inserts step 2
Software Citation vs Paper Citation (cont.)

- Software citation principles guidance may not work
  - Adds a step to the software developers workflow
  - They may not care enough to implement it
- Even if we do get to a future time in which developers routinely published their software releases, what happens until then, or for existing software?
- Real problem:
  - Steps (create, publish, cite) don’t match how open source is developed and used
    - Software is more fine-grained and iterative
    - Open source development mostly occurs in the open
  - No natural need for publish step, other than marketing and credit, which are not primary concerns in all projects
Software Citation vs Paper Citation (cont.)

- Back to papers: what happens if the citer wants to refer to something that has not been published?
  - Students initially taught to avoid this situation, later taught to cite as “personal communication”
  - APA Publication Manual distinguishes between recoverable and unrecoverable data.
  - Recoverable data (that which can be accessed by the reader via the citation information) should be cited as a formal citation.
  - Unrecoverable data should be referred to within the text as “(author, personal communication, date)”
- This distinction between recoverable (published) and unrecoverable (not available) doesn’t work for software.
- All versions of software on GitHub, even if never published, are recoverable by default.
  - Unless project is deleted from GitHub; could still be recovered from a local copy.
- Regarding credit, Software Citation Principles paper: “It is not that academic software needs a separate credit system from that of academic papers, but that the need for credit for research software underscores the need to overhaul the system of credit for all research products.”
- The fact that the three-step model of distinct creator, publisher, and citer doesn’t really fit modern open source practices is another argument for that overhaul.

Working group status & next steps

- Principles document published in PeerJ CS
- Software Citation Working Group (co-chairs Smith, Katz, Niemeyer) ends
  
  we are here now!

- Software Citation Implementation group (co-chairs Katz, Fenner, Chue Hong) starts
- Planning…
  - Work with institutions, publishers, funders, researchers, etc.,
  - Considering endorsement period for both individuals and organizations
    - Want to endorse? Email/talk to me
    - Write full implementation examples paper?
- Want to join? Sign up on new FORCE11 group page
  - https://www.force11.org/group/software-citation-implementation-working-group