Information Services & Use 41 (2021) 39–42
DOI 10.3233/ISU-210108
IOS Press

Software citation

Daniel S. Katz[a,*] and Shelley Stall[b]
[a]*Chief Scientist at the National Center for Supercomputing Applications (NCSA) and Research Associate Professor in Computer Science, Electrical and Computer Engineering, School of Information Sciences (iSchool), University of Illinois Urbana-Champaign, Urbana, IL, USA*
[b]*Senior Director, Data Leadership Program, American Geophysical Union, Washington, DC, USA*

Abstract. Nearly all research today has a digital component, and typically, scholarly results are strongly dependent on software. For the research results to be fully understood, the software that is used must be uniquely identified. Research software is frequently developed by researchers themselves, often initially to solve a single problem, and then later generalized to solve additional problems. Ideally, the software is shared so that other researchers can also benefit and avoid the duplicate work required for development and maintenance. The researchers must expect and receive value for their contribution and sharing. Because publishing is a key element of our existing scholarly structures, the research that was done must be clearly explained in papers. This can be used to create incentives for researchers not only to share their software, but also to contribute to community software, in both cases through software citation. Contributors to software that is used in papers and is cited by those papers can become authors of the software as it is tracked by indexes, which also track how often the software is cited.

Keywords: Software citation, scholarly publishing, citation indexing, journals, bibliometrics

1. Introduction

Nearly all research today has a digital component. Examples include the software and services that are used to: (1) write manuscripts and prepare visualizations; (2) create, collect, clean, organize, and analyze data such as text analysis, survey results, or telescope images; (3) simulate processes such as global climate or nuclear physics; and (4) optimize engineering designs, etc. While some of the software used in these processes is, in some sense, a commodity, meaning that the choice of the specific software product does not make a difference in the scholarly conclusions, in most cases, the scholarly results are strongly dependent on the software, or rather, the (research) software is an essential element of the research. For the research results to be fully understood, the software that is used must be uniquely identified.

Research software is often developed by researchers themselves, often initially to solve a single problem, and then later generalized to solve other problems, and ideally shared so that other researchers can also benefit and avoid duplicate work required for development and maintenance. When this process works, the software can become community software, developed, supported, and maintained by a community of user developers, expanding as it is applied to new problems. Whether or not this happens strongly depends on both community culture and associated incentives. The researchers must expect and receive value for their contribution and sharing. The users expect well-managed and documented software.

Publishing is a key element of our existing scholarly structures: researchers are rewarded for publishing their own research as well as for having their published research be cited. And increasingly, publishers and scholarly communities are pushing for these papers to be as transparent and as reproducible as possible.

*Corresponding author: Daniel S. Katz. E-mail: d.katz@ieee.org.

This requires that the research that was done is clearly explained. Both publishing and reproducibility can be used to create incentives for researchers to share their software, and to contribute to community software, in both cases through software citation. Contributors to software that is used in papers and is cited by those papers can become authors of the software as it is tracked by indexes that also track how often it is cited. And the need for reproducibility also leads to a need to cite the exact version of the software that was used in the research.

Software citation improves scholarship by making the software that was used clear and improves incentives for sharing and contributing to software by making software a cited artifact. This is a relatively new idea, not yet fully-integrated into the broad community involved with scholarly communication. Common practices are being developed at multiple levels, and here, we focus on two of them: (1) work with publishers to set expectations and provide guidance to their authors, reviewers, and editors so that software citations are added to papers, and (2) work with publishers and their production processes to ensure that those software citations are machine-readable throughout the scholarly publishing ecosystem.

2. Software citation — common journal guidance for authors, reviewers, and editors

Researchers are adopting better transparency and reproducibility practices and including mentions of the software significant to their research in their papers. Until recently, there has been little guidance available to journal staff to help authors with recommended practices for software citation. The FORCE11 Software Citation Implementation Working Group (see: https://www.force11.org/group/software-citation-implementation-working-group) has recently brought together a task force that includes journal staff, software developers, and representatives of the scholarly infrastructure to document common guidelines. These guidelines are built upon prior work of the FORCE11 working group, including guidelines for authors [1] and developers [2].

The journal guidelines provide a recommended list of citation elements as well as use cases and examples [3]. The new guidance was recently highlighted in a *Scholarly Kitchen* article that focuses on the value of citing research software for reasons such as reproducibility, reuse, and credit [4]. All journals are invited to adopt the practices in their own author guidelines, which includes providing suitable examples in the citation style for the journal.

3. Publisher processes supporting software citation

Software citations take two forms during the journal production process. The first is what the authors view online when they edit their proof version that will later be rendered on the journal website and possibly printed on physical paper. In their manuscript, the author may include information about the software, possibly along with pseudocode, in the methods and data sections. The author describes how to access the software in the availability statement, with information on location, fees, or contact details for proprietary software. However, in order for the author to give automated attribution and credit to the software creators, there must be a software citation in the references section. This is where the machine-readable version of the citation can diverge from the human-readable version and possibly become inaccurate. This inaccuracy can result in problems with the elements that are needed for automated attribution and credit.

The journal production process, including copy editing, review of the final proof of the paper, and the final steps to publication, has both manual and automated steps to ensure that all the elements of the paper

are reviewed and linked. Different journals have somewhat different approaches, but all journals ensure that the citations are as accurate as possible.

It is critical that the broad journal community review the quality of both the human readable online publications as well as the machine-readable version to support automated credit and attribution. The FORCE11 Software Citation Implementation Working Group's Journal Task Force is developing guidance that journals can use to correct these difficult-to-identify changes to the machine-readable version of software citations.

Even more prominently, encouraging an author to cite the software they used in the references section continues to be a challenge. Software packages that are important intellectual contributions to the discipline tend to only be mentioned in the text of the manuscript. Without the citation, linking the software to the paper doesn't happen in the journal workflow. The citation in academia equates to the importance of the software to the research. As the research ecosystem better recognizes output beyond the paper publication, these software citations are encouraged. Organizations keen to recognize important software packages develop both manual methods and artificial intelligence to create the connections. The NASA Astrophysics Data System (ADS) in partnership with the American Astronomical Society, funded by the Alfred P. Sloan Foundation in their Asclepias Project [5], is an example of a discipline investing in the bibliographic indexing of software with links to the publications that are important to the community.

The Asclepias Project objectives are viewed by the astrophysics community as beneficial to better understanding the software that is integral to their research findings. Other disciplines can similarly benefit significantly by taking simple steps to connect important software by citing the preserved version in the references section of the author's paper.

Complementary to this work, the JATS4R (see: https://jats4r.org) community has recently started working on guidance around software citation intended to align with the use cases identified in the FORCE11 journal guidance as well as the new Crossref schema that will be announced later this year.

Through these communities, NISO, and the broad journal community, we want to encourage authors to cite their relevant software and encourage journals to capture and validate these citations in a way that supports credit to the software creators.

About the authors

Daniel Katz (corresponding author) is Chief Scientist at the National Center for Supercomputing Applications (NCSA) and Research Associate Professor in Computer Science, Electrical and Computer Engineering, and the School of Information Sciences (iSchool), at the University of Illinois Urbana-Champaign. In past lives, he worked in other universities (Chicago, Louisiana State), government (NSF, JPL), and industry (Cray Research), all focusing on software in some aspect. He still does many things related to software in a variety of organizations and roles, mostly aimed at making software better and providing credit and recognition to its developers. Email: d.katz@ieee.org; Phone: 217 244 8000.

Shelley Stall is the Senior Director for the American Geophysical Union's Data Leadership Program. She works with AGU's members, their organizations, and the broader research community to improve data and digital object practices with the ultimate goal of elevating how research data is managed and valued. Shelley's diverse experience working as a program and project manager, software architect, database architect, performance and optimization analyst, data product provider, and data integration architect for international communities, both nonprofit and commercial, provides her with a core capability to guide

development of practical and sustainable data policies and practices ready for adoption and adapting by the broad research community. Email: sstall@agu.org.

References

[1] N.P. Chue Hong, A. Allen, A. Gonzalez-Beltran, A. de Waard, A.M. Smith et al., *Software Citation Checklist for Authors* (Version 0.9.0), (2019, October 15), Zenodo, doi:10.5281/zenodo.3479199, accessed July 11, 2021.

[2] N.P. Chue Hong, A. Allen, A. Gonzalez-Beltran, A. de Waard, A.M. Smith et al., *Software Citation Checklist for Developers* (Version 0.9.0), (2019, October 15), Zenodo, doi:10.5281/zenodo.3482769, accessed July 11, 2021.

[3] D.S. Katz, N.P. Chue Hong, T. Clark, A. Muench, S. Stall et al., *Recognizing the Value of Software: A Software Citation Guide* [version 2; peer review: 2 approved], F1000Research 9:1257, 2021, doi:10.12688/f1000research.26932.2, accessed July 11, 2021.

[4] D.S. Katz and H. Murray, Guest Post – *Citing Software in Scholarly Publishing to Improve Reproducibility, Reuse, and Credit,* (2021, January 21), https://scholarlykitchen.sspnet.org/2021/01/21/guest-post-citing-software-in-scholarly-publishing-to-improve-reproducibility-reuse-and-credit/, accessed July 11, 2021.

[5] American Astronomical Society Publishing, Updates to Software Citation in NASA ADS and Zenodo, (2019, January 10), https://journals.aas.org/news/asclepias_aas233/, accessed July 11, 2021.