

# Using citation to provide credit for software contributions

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

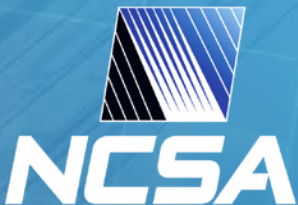
Helmholtz Open Science Webinar

9 & 17 August 2017



HELMHOLTZ  
| GEMEINSCHAFT

Open Science



National Center for Supercomputing Applications  
University of Illinois at Urbana–Champaign

# Software in research

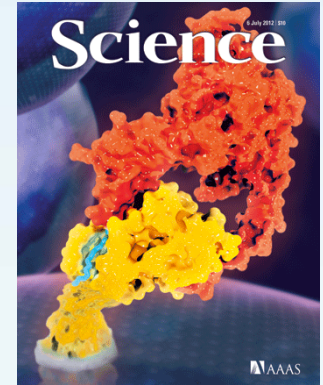
- Claim: software (including services) essential for the bulk of research
- Evidence from surveys
  - UK academics at Russell Group Universities (2014)
  - Members of (US) National Postdoctoral Research Association (2017)
  - My research would not be possible without software: 67% / 63% (UK/US)
  - My research would be possible but harder: 21% / 31%
  - It would make no difference: 10% / 6%

S. Hettrick, “It’s impossible to conduct research without software, say 7 out of 10 UK researchers,” Software Sustainability Institute, 2014. Available at: <https://www.software.ac.uk/blog/2016-09-12-its-impossible-conduct-research-without-software-say-7-out-10-uk-researchers>

S.J. Hettrick, M. Antonioletti, L. Carr, N. Chue Hong, S. Crouch, D. De Roure, et al, “UK Research Software Survey 2014”, Zenodo, 2014. doi: 10.5281/zenodo.14809.

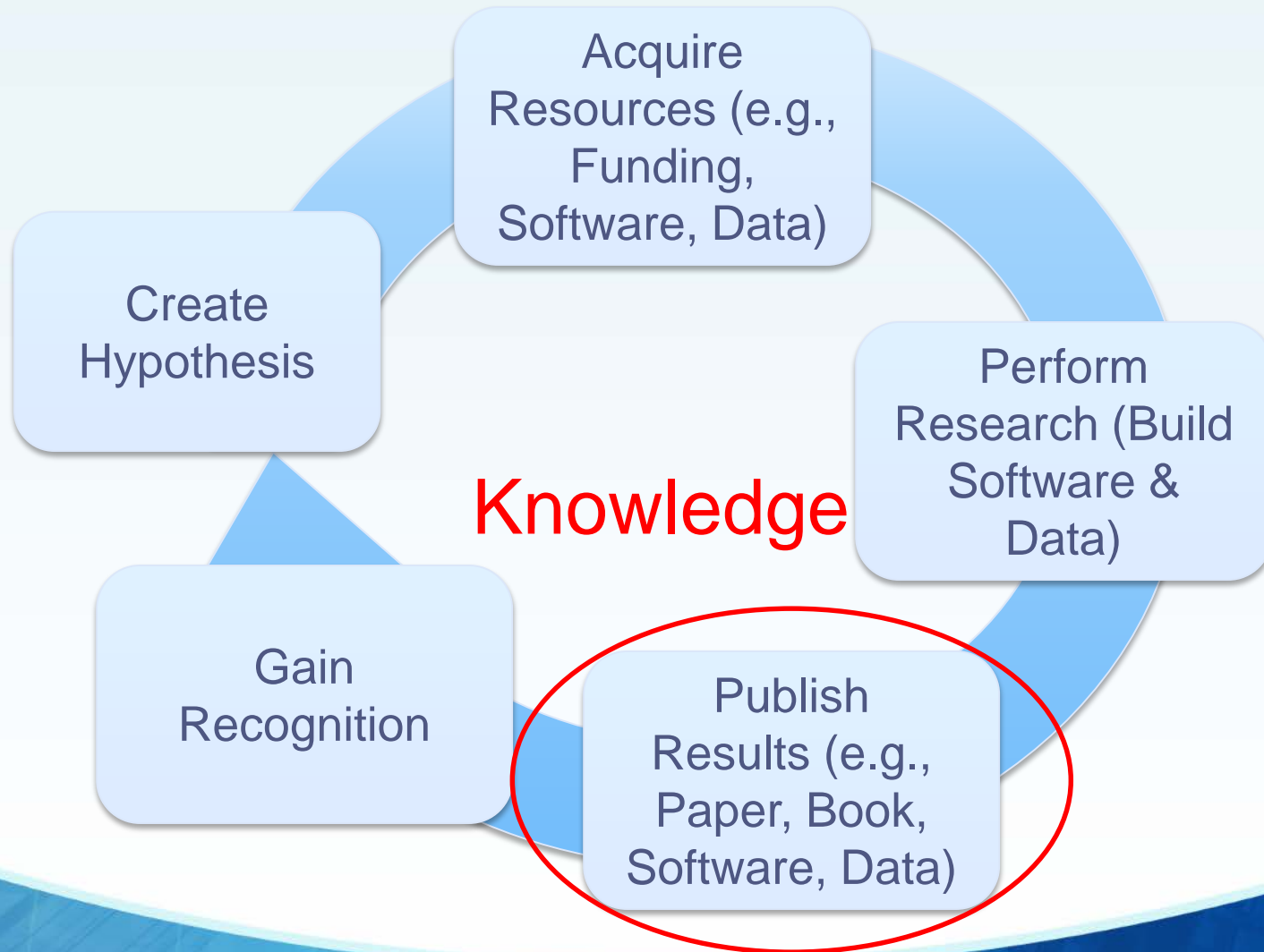
U. Nangia and D. S. Katz, “Track 1 Paper: Surveying the U.S. National Postdoctoral Association Regarding Software Use and Training in Research,” Zenodo, 2017. doi: 10.5281/zenodo.814102

# Software in scholarship



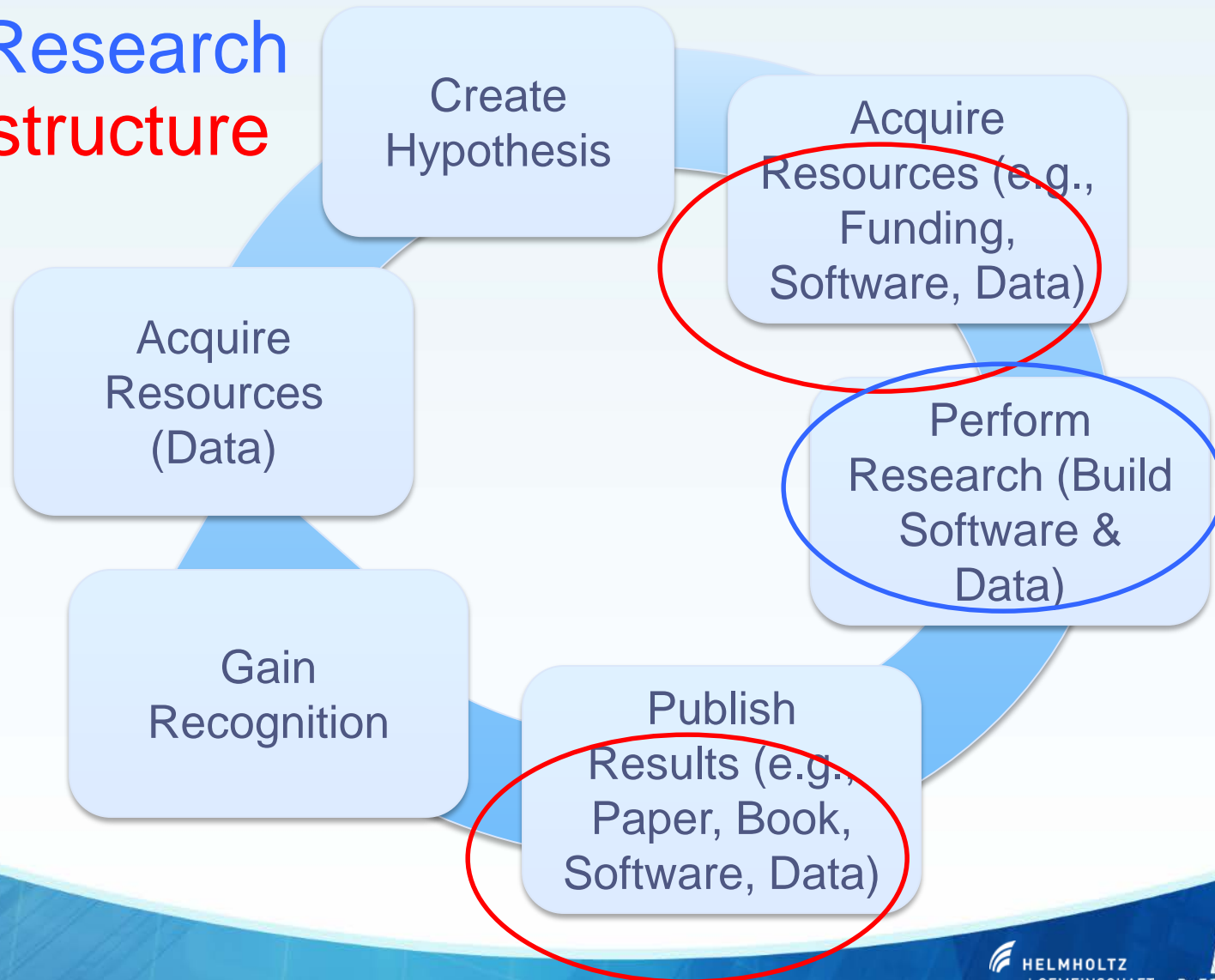
- Claim: software (including services) essential for the bulk of research
- Evidence from journals:
  - About half the papers in recent issues of Science were software-intensive projects
  - In Nature Jan–Mar 2017, software mentioned in 32 of 40 research articles
    - Average of 6.5 software packages mentioned per article

# Computational science research



# Purposes of software in research

## Research Infrastructure



# Research software vs. infrastructure software

- Some software is intended for research
  - Funded by many agencies, sometimes explicitly, often implicitly
  - Intended for immediate use by developer
  - Maybe archived for future use and reproducibility
- Other software is intended as infrastructure
  - Funded by some agencies, almost always explicitly
  - Intended for use by community (not just developer)
- Focus mostly on infrastructure software, but many issues cross between
  - Reproducibility causes the most overlap



# How to better measure software contributions

- Citation system was created for papers/books
- We need to either/both
  1. Jam software into current citation system
  2. Rework citation system
    - Focus on 1 as possible; 2 is very hard.
- Challenge: not just how to identify software in a paper
  - **How to identify software used within research process**
- Howison: random sample of 90 articles in the biology literature -> 286 software mentions in 7 different ways (e.g., cite software paper, cite user manual, cite name, URL in text)

# Software citation principles: People & Process

- FORCE11 Software Citation group started July 2015 (co-leads Smith & Katz)
- WSSSPE3 Credit & Citation working group joined September 2015 (Niemeyer joined as co-lead)
- ~60 members (researchers, developers, publishers, repositories, librarians)
- Work on GitHub <https://github.com/force11/force11-scwg> & FORCE11 <https://www.force11.org/group/software-citation-working-group>
- 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 (with reviews)
  - Smith AM, Katz DS, Niemeyer KE, FORCE11 Software Citation Working Group.(2016) Software Citation Principles. *PeerJ Computer Science* 2:e86. DOI: [10.7717/peerj-cs.86](https://doi.org/10.7717/peerj-cs.86) and <https://www.force11.org/software-citation-principles>
- Contains: principles (general statements), use cases (where the principles should apply), discussion (suggestions on how to apply principles)



# 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.



# 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

# Journal of Open Source Software (JOSS)

- In the meantime, there's JOSS
- A developer friendly journal for research software packages
- “If you've already licensed your code and have good documentation then we expect that it should take **less than an hour** to prepare and submit your paper to JOSS”
- Everything is open:
  - Submitted/published paper: <http://joss.theoj.org>
  - Code itself: where is up to the author(s)
  - Reviews & process: <https://github.com/openjournals/joss-reviews>
  - Code for the journal itself: <https://github.com/openjournals/joss>
- Zenodo archives JOSS papers and issues DOIs
- First paper submitted 4 May 2016
  - 31 May 2017: 111 accepted papers, 41 under review, ~15 submitted (pre-review)
  - 31 July 2017: 125 accepted papers, 35 under review, ~30 submitted (pre-review)

# Working group status & next steps

- Software Citation Working Group (co-chairs Smith, Katz, Niemeyer) ended March 2017
- Software Citation Implementation group (co-chairs Katz, Fenner, Chue Hong) started (slowly) March 2017
- Now 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>

# Additional material

# Software vs. data in the context of citation

- Software is data, but it is not just data
- Data (in computing and information science): anything that can be processed by a computer
- Software: special kind of data that can be a creative, executable tool that operates on data
- Software & data are similar in with regard to credit and metrics, and both traditionally have not been cited in publications



# 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.

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