Software Citation Implementation

FORCE11

The Future of Research Communications and e-Scholarship

Daniel S. Katz, Martin Fenner, Neil Chue Hong
Working group co-chairs

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

ILLINOIS

NCSA | National Center for Supercomputing Applications

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%

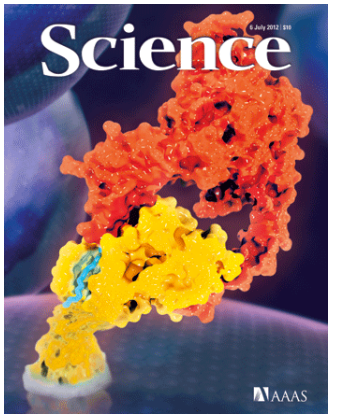
S. Hettrick, "It's impossible to conduct research without software, say 7 out of 10 UK researchers," Software Sustainaiblity Institute, 2014. Available at: https://www.software.ac.uk/blog/2016-09-12-its-impossible-conduct-research-without-software-say-7-out-10-uk-researchers

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," WSSSPE5.1, 2017. doi: 10.6084/m9.figshare.5328442.v1

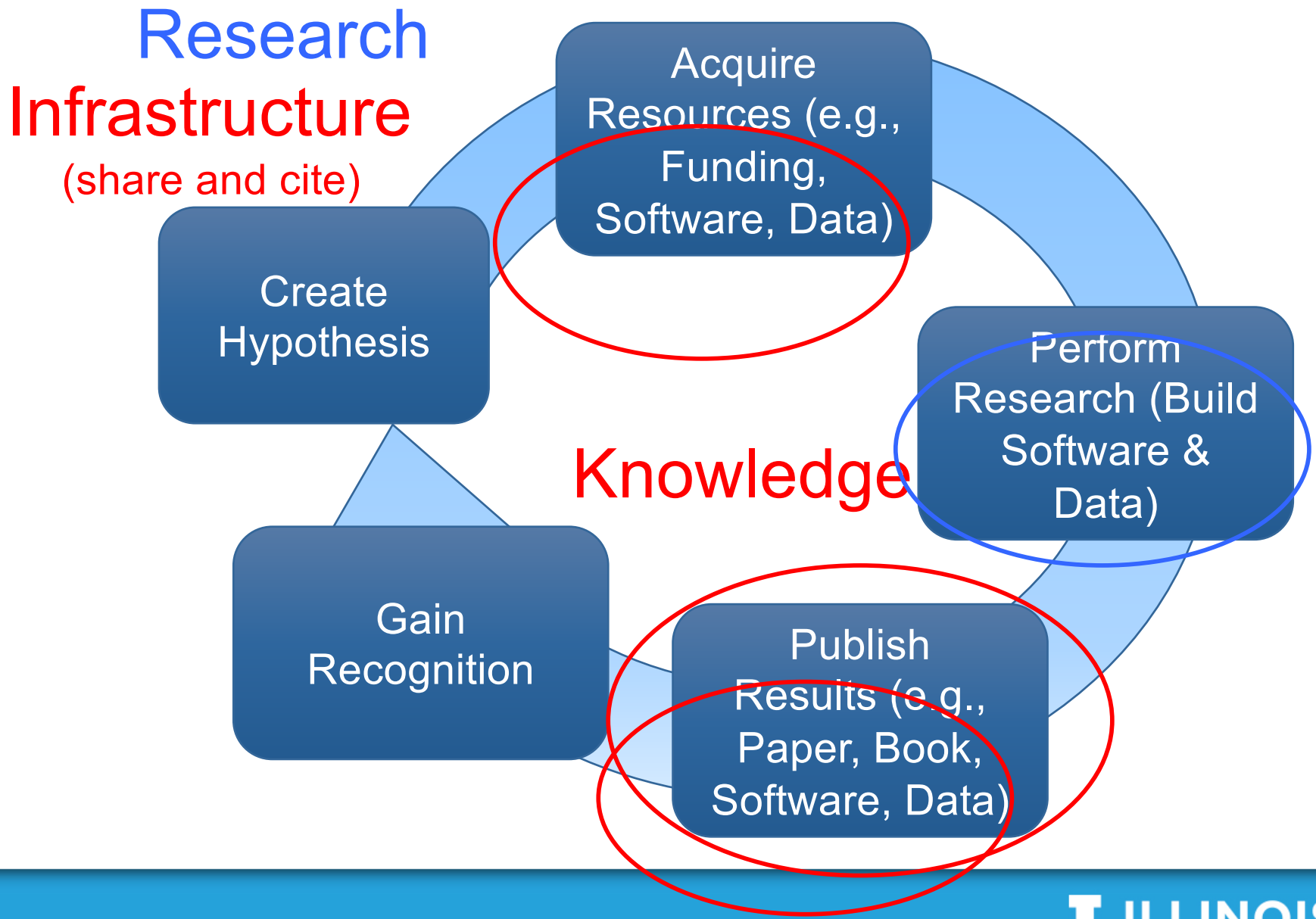
Software in scholarship

- 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

Software in research cycle



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**

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)
- Working 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
  - Discussion via GitHub issues, changes tracked
  - Contents: 6 principles, discussion, use cases, …
- Submitted, reviewed and modified, published
  - 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 and https://www.force11.org/software-citation-principles
  - Also includes reviews and responses

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.

Discussion Example: 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

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

- 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

Working group status

- Principles document published in PeerJ CS
- Software Citation Working Group ended
- Software Citation Implementation group now in progress
  - Goal is implementing software citation
  - Working with institutions, publishers, funders, researchers, etc.

Paper citation

- Three relevant steps for paper citation
  1. Creator (aka author) submits paper and metadata 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

Software citation

- For open source software today
  - Creator develops software on GitHub, released at different stages (versions) during its development
  - Someone who uses that software may not cite it; 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

- Software citation principles guidance is of limited value
  - Principles themselves still seem good
  - But technically how to implement them is not clear
  - Software citation principles guidance adds a step to the open source software developer's workflow
  - They may not care enough to implement it
- Real problem for open source:
  - 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
- Also, we don't address non-open source software sufficiently
- And social challenges around adoption are another step

ILLINOIS

NCSA | National Center for
Supercomputing Applications