2nd Conference 2017 on Non-Textual Information

"Software and Services for Science (S3)"

May 10-11, 2017 in Hannover

TIB

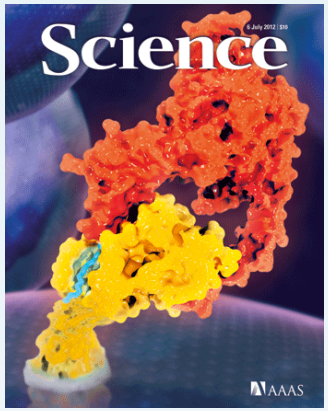
Software citation: a cornerstone of software-enabled research

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

NCSA

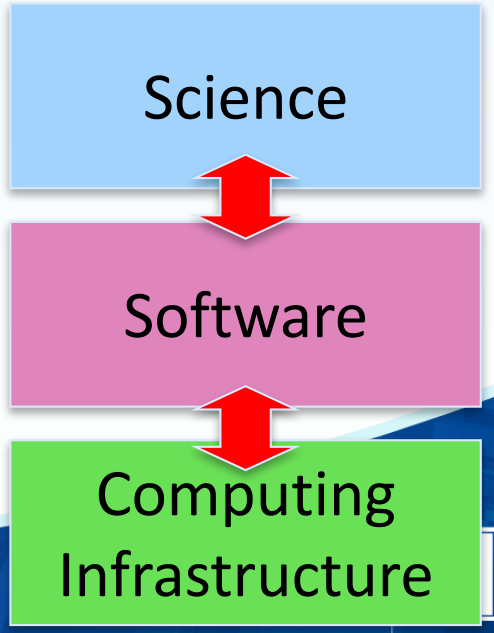
National Center for Supercomputing Applications
University of Illinois at Urbana–Champaign

Software as infrastructure

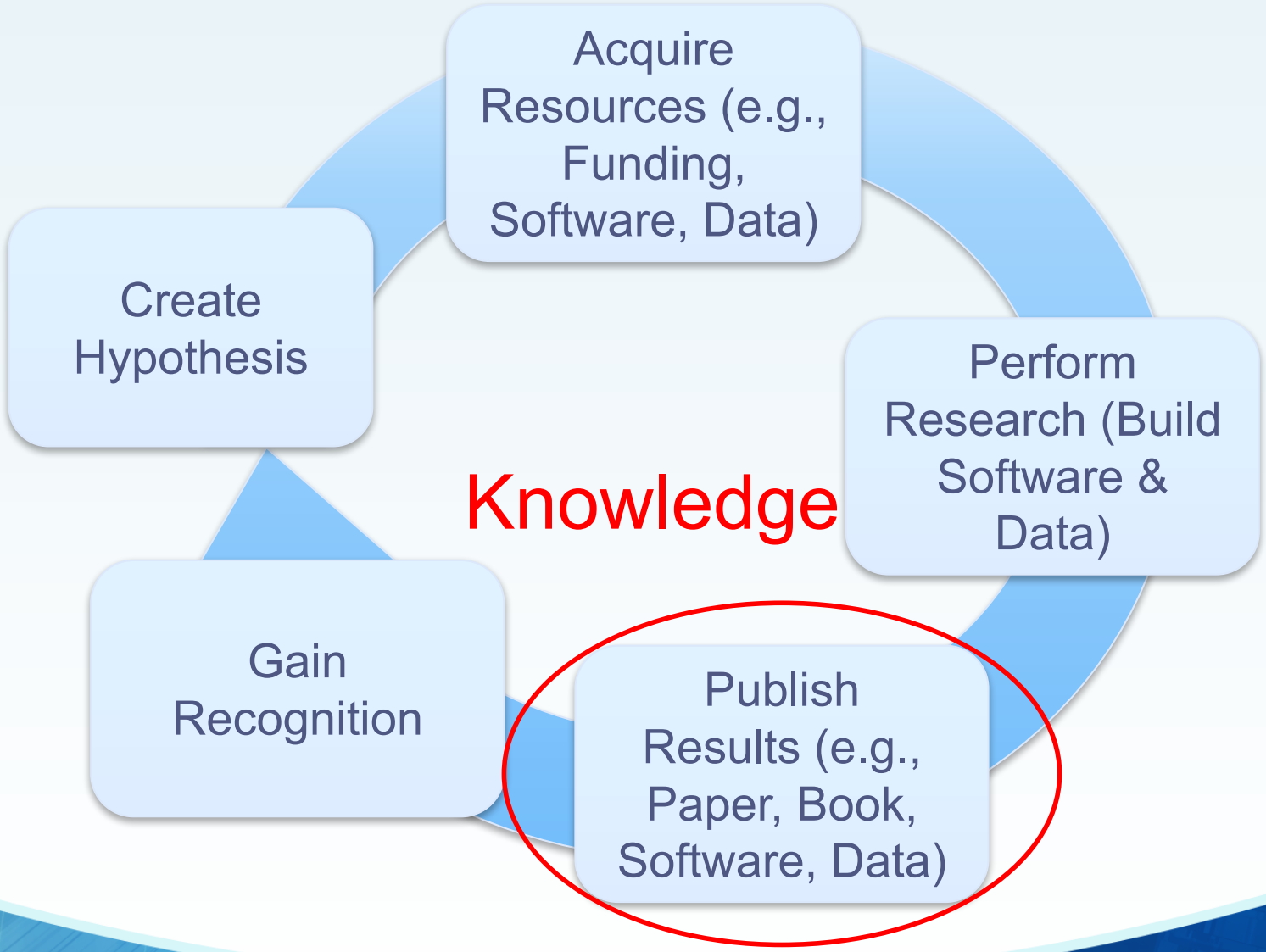


- Software (including services) essential for the bulk of science
  - About half the papers in recent issues of Science were software-intensive projects
  - Research becoming dependent upon advances in software
  - Wide range of software types: system, applications, modeling, gateways, analysis, algorithms, middleware, libraries

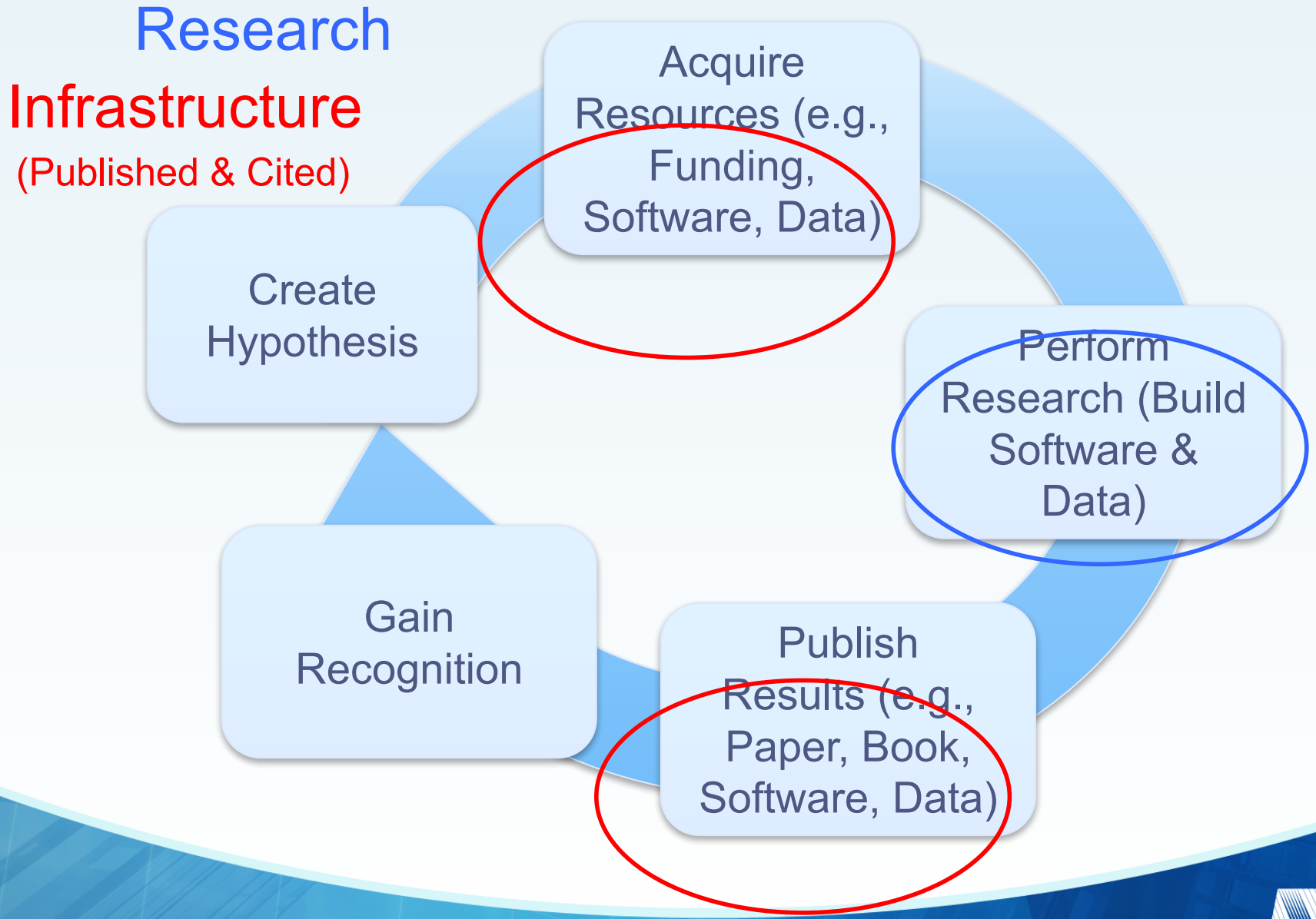
- Software is not a one-time effort, it must be sustained
  - Development, production, and **maintenance** are people intensive
  - Software life-times are long vs hardware
  - Software has under-appreciated value



Computational science research



Purposes of software in research



Software Citation Motivation

- Scientific research is becoming:
  - More open – scientists want to collaborate; want/need to share
  - More digital – outputs such as software and data; easier to share
- Significant time spent developing software & data
- Efforts not recognized or rewarded
- Citations for papers systematically collected, metrics built
  - But not for software & data
- Hypothesis:

  Better measurement of contributions (citations, impact, metrics)

  —> Rewards (incentives)

  —> Career paths, willingness to join communities

  —> More sustainable software

How to better measure software usage

- 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 today

- Software and other digital resources currently appear in publications in very inconsistent ways
- Howison: random sample of 90 articles in the biology literature -> 7 different ways that software was mentioned

Mention Type	Count (n=286)	Percentage
Cite to publication	105	37%
Cite to users manual	6	2%
Cite to name or website	15	5%
Instrument-like	53	19%
URL in text	13	5%
In-text name only	90	31%
Not even name	4	1%

- Studies on data and facility citation -> similar results

J. Howison and J. Bullard. Software in the scientific literature: Problems with seeing, finding, and using software mentioned in the biology literature. Journal of the Association for Information Science and Technology, 2015. In press. http://dx.doi.org/10.1002/asi.23538.

NCSA

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)
- 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
    - Katz DS, Niemeyer KE, et al (2016) Software vs. data in the context of citation. PeerJ Preprints 4:e2630v1. DOI: 10.7287/peerj.preprints.2630v1
  - Discussion via GitHub issues, changes tracked
- Submitted, reviewed and modified (many times), now 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

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
- https://github.com/force11/force11-scwg
- 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.

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

Purdue Online Writing Lab. Reference List: Electronic Sources (Web Publications). https://owl.english.purdue. edu/owl/resource/560/10/, 2015.

NCSA

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: Unique identification

- 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: 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

APA Publication Manual: http://www.apastyle.org/manual/index.aspx

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

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 May 4, 2016
  - As of 18 January: 62 accepted papers, 21 under review, 17 submitted (pre-review)

Conclusions

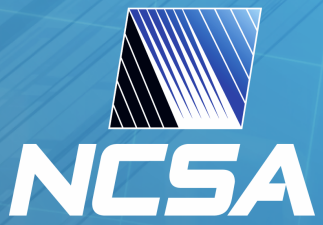
- Software
  - Important today, essential tomorrow
- Citation
  - We know what to do (mostly), now need to do it
- What you can do
  - Cite the software you use, make it easy for others to cite the software you write (and see the “I solemnly pledge” manifesto – http://ceur-ws.org/Vol-1686/WSSSPE4_paper_15.pdf)

Credits

- Thanks to Arfon Smith and Kyle Neimeyer for co-leadership in FORCE11 Software Citation WG
- More of my thinking about this:
  - Blog: http://danielskatzblog.wordpress.com
  - Tweets: @danielskatz

Software citation: a cornerstone of software-enabled research

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



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