Recent developments and initiatives in scholarly publishing

Alberto Accomazzi
IVOA Interop, Cape Town, South Africa
12 May 2016

Overview

- Use of API and ORCIDs in ADS
- The Unified Astronomy Thesaurus
- Annotating All Knowledge Coalition
- Authorship and Contributor Roles
- Software Citation Principles
- Software Publishing
- Places where this stuff is discussed

API access tied to Bumblebee accounts

ADS API access tied to Bumblebee accounts

ADS API Usage

kpub: Kepler publication data

A database of scientific publications related to NASA's Kepler/K2 m

kpub is a mission-specific tool that enables NASA's Kepler/K2 Guest O
scientific publications in an easy way. It leverages SQLite and the ADS
create and curate a database that contains the metadata of mission-rel

Example use

Print a nicely-formatted list of Kepler-related exoplanet publications in r

```
kpub --exoplanets
```

Add a new article to the database using its bibcode. This command will
classify the science:

```
kpub-add 2015arXiv150204715F
```

Brendan Griffen
Astrophysicist, Massachusetts Institute of Technology

blog   projects   about   vitae   contact

A Brief Analysis of Hubble, Einstein and Sagan Fellows With ADS

04 April 2016

Now that the job hunt is largely over, I thought it might be interesting to briefly examine the publication profiles of previous and current holders of fellowships which are the most in demand. The core fellowships I focus on are the NASA-funded Einstein, Hubble and Sagan fellowships which often get hundreds of applicants each year.

ORCID Claiming and Indexing in ADS

Claiming since 1/1/2016:

- 917 users
- 50,011 papers
- 81,000 unique claims

Record indexing as of 4/1/2016:

- 14,500 publisher records
- 48,942 claimed+validated records
- 62,934 records with at least one ORCID mapping

Unified Astronomy Thesaurus v.1 is here!

Today I am releasing version one of the Unified Astronomy Thesaurus (UAT v.1).

The UAT has been completely overhauled; restructured into new top level categories and re-organized throughout. There have been many major revisions to bring it more inline with the way astronomers and astrophysicists study the universe.

I want to thank Sarah Weissman, Josh Peek, Kayleigh Bohemier, Dianne Dietrich, Jane Holmquist, Barbara Kern, and especially Jill Lagerstrom for all of the work each of you put into revising and updating the thesaurus. I also want to thank the many researches and scientists who lent their expertise to this project. Because of all of you, every term in the UAT was looked at, revised, edited, tweaked, or moved.

Version 1 of the Unified Astronomy Thesaurus has 1834 terms, 11 top level categories, a depth of 10 terms, and 319 'related term' links. For comparison, the beta version of the UAT had 1920 terms, 15 top level categories, a depth of 15 terms, and 224 'related term' links.

In addition to the major restructuring of the UATs top level categories and overall organizational structure, 321 terms were removed, 236 new terms were added, and 95 new 'related term' links were added.

Recent Posts

- Unified Astronomy Thesaurus v.1 is here! December 23, 2015
- Update on the UAT September 11, 2014
- The UAT at the e-Sciences Symposium April 10, 2014
- What's new with the Unified Astronomy Thesaurus February 10, 2014
- Download the UAT September 10, 2013

Annotating All Knowledge Coalition

Group of publishers, information providers, tech firms announce plans to work together towards open, portable annotations based on the Hypothes.is platform. This includes Wiley, CrossRef, PLOS, Project Jupyter, HighWire, arXiv and ADS

http://www.nature.com/news/annotating-the-scholarly-web-1.18900

https://hypothes.is/annotating-all-knowledge/

https://hypothes.is/for-publishers/

Envisioning a Scholarly Annotation Layer

- It must be built on an open but standard framework that enables global discovery and community enrichment.
- It must support granular annotation of elements in all key formats, and across different representations of the same content (e.g. PDF vs HTML).
- There must be a diversity of interoperable annotation systems.
- These systems must be fully accessible to humans, and machines that will use APIs to create and mine annotations.
- It must be possible to identify people, groups, and resources in global ways, so that sharing, discovery, and interconnection can span repositories and annotation services.

Authorship and Contributor Roles

The Contributor Roles Taxonomy project (Project CRediT) emerged to address recognition that the concept of 'authorship' in producing scientific scholarly output is outdated and no longer fit for purpose. Project CrediT aims to provide transparency to the contributions of researchers to scholarly published work, to enable discoverability and to improve attribution, credit, and accountability.

Contributor roles currently used by: Cell, PLOS, Nature. Is this the long-term destiny for all?

http://casrai.org/credit
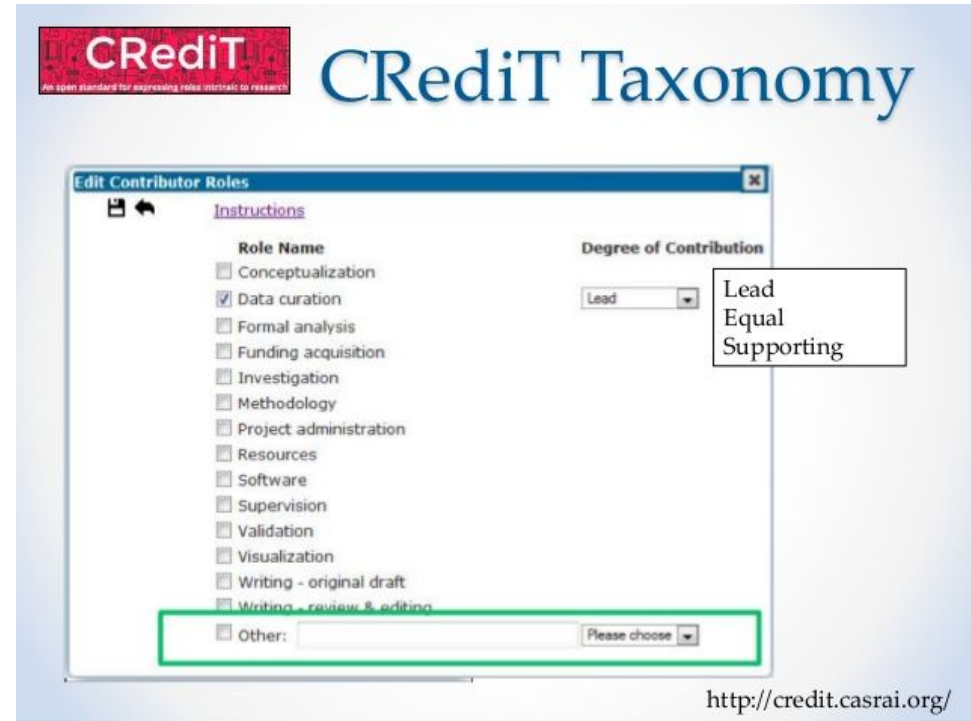
Software Citation Principles

Published in April 2016, follows the data citation principles:

- Importance
- Credit and Attribution
- Unique Identification
- Persistence
- Accessibility
- Specificity

https://www.force11.org/software-citation-principles

SOFTWARE CITATION PRINCIPLES

FORCE11 SOFTWARE CITATION WORKING GROUP (EDITORS: ARFON M. SMITH, DANIEL S. KATZ, KYLE E. NIEMEYER)

Abstract. Software is a critical part of modern research and yet there is little support across the scholarly ecosystem for its acknowledgement and citation. Inspired by the activities of the FORCE11 working group focussed on data citation, this document summarizes the recommendations of the FORCE11 Software Citation Working Group and its activities between June 2015 and April 2016. Based on a review of existing community practices, the goal of the working group was to produce a consolidated set of citation principles that may encourage broad adoption of a consistent policy for software citation across disciplines and venues. Our work is presented here as a set of software citation principles, a discussion of the motivations for developing the principles, reviews of existing community practice, and a discussion of the requirements these principles would place upon different stakeholders. Working examples and possible technical solutions for how these principles can be implemented will be discussed in a separate paper.

1. SOFTWARE CITATION PRINCIPLES

The principles in this section are written fairly concisely, and discussed further later in this document (§5). Here, for example, we do not define what software should be cited, but how it should be cited, and we talk about how such decisions might be made in the discussion section (§5).

(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 products such as papers or books, that is, authors should cite the appropriate set of software products just as they cite the appropriate set of papers.

(2) **Credit and Attribution**: Software citations should facilitate giving scholarly credit and normative and legal attribution to all contributors to the software, recognizing that a single style or mechanism of attribution may not be applicable to all software.

(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.

(4) **Persistence**: Unique identifiers and metadata describing the software and its disposition should persist – even beyond the lifespan of the software they describe.

(5) **Accessibility**: Software citations should permit and 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.

(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.

These software citation principles were originally based on an adaptation of the FORCE11 Data Citation Principles [11], and then were modified based on discussions of the FORCE11 Software Citation Working Group (see Appendix A for members), information from the use cases in §3, and the related work in §4. The adaptations have been made because software, while similar to data in terms of not traditionally having been cited in publications, is also different than data in that it can be used to express or explain concepts, it is updated more frequently, and it is executable. Also, while software can be considered a type of data, the converse is not generally true.

Software Publishing

Now many options available for "[pP]ublishing" and taking credit for Software:

- Put it on Github/Bitbucket/<insert favorite repo here> with a CITATION file specifying how you would like to be acknowledged
- Make a release and push the code to a preservation system such as Zenodo (DOI is minted and code release is preserved)
- Write a minimal software paper in a publication such as the Journal of Open Source Software: http://joss.theoj.org (code review is applied)
- Write a full paper describing the software in an appropriate journal such as Astronomy & Computing, A&A or in AAS journals (new - see http://journals.aas.org/policy/software.html)

The Journal of Open Source Software

A **developer friendly** journal for research software packages.

Learn more »

☐ **hrbrmstr / metricsgraphics** JOSS Under Review

MetricsGraphics.js is a library built on top of D3 that is optimized for visualizing and laying out time-series data. It provides a simple way...

10.5281/zenodo.50373 ☐

☐ **diana-hep / carl** JOSS Under Review

Carl is a toolbox for likelihood-free inference in Python.

10.5281/zenodo.47798 ☐

Astronomy and Computing

Supports Open Access

(i) Software Release Papers. To be suitable for publication in the journal, these should do more than just describe a new or updated software package. They should emphasize innovative factors like the intellectual contribution represented by a new algorithm or the use of a new technology, and should make clear the ways in which the software is of significant value to the community. The editors anticipate that software releases that merit a journal publication will be professionally packaged and documented, and made available from a stable URL, preferably with the source code available in a public repository: (see section below on "Source code repositories").

Policy Statement on Software

1 January 2016

AAS Journals have adopted a policy that reflects the importance of software to the astronomical community, and the need for clear communication about such software which ensures that credit is appropriately given to its authors. The policy provides clear guidelines for citing software in all papers, and supports the publication of descriptive papers about software relevant to research in astronomy and astrophysics.

Guidelines for software papers

AAS Journals welcome papers which describe the design and function of software of relevance to research in astronomy and astrophysics. Such papers should contain a description of the software, its novel features and its intended use. Such papers need not include research results produced using the software, although including examples of applications can be helpful. There is no minimum length requirement for software papers.

If a piece of novel software is important to published research then it is likely appropriate to describe it in such a paper.

Reproducible Research: Citing your execution environment using Docker and a DOI

By Robert Haines, Institute Fellow & Research Software Engineering Manager, IT Services, University of Manchester and Caroline Jay, Institute Fellow & Lecturer, School of Computer Science, University of Manchester.

As we move into a world where (hopefully) more and more people are trying to make their research as reproducible as possible, a lot of us are turning to Docker to help out with the task of distributing our research software in a way in which it is as accessible as possible to others. As we move in this direction we need to be able to cite the software environments that we are executing, not just the source code itself.

Image courtesy of Andy Brown, BBC R&D
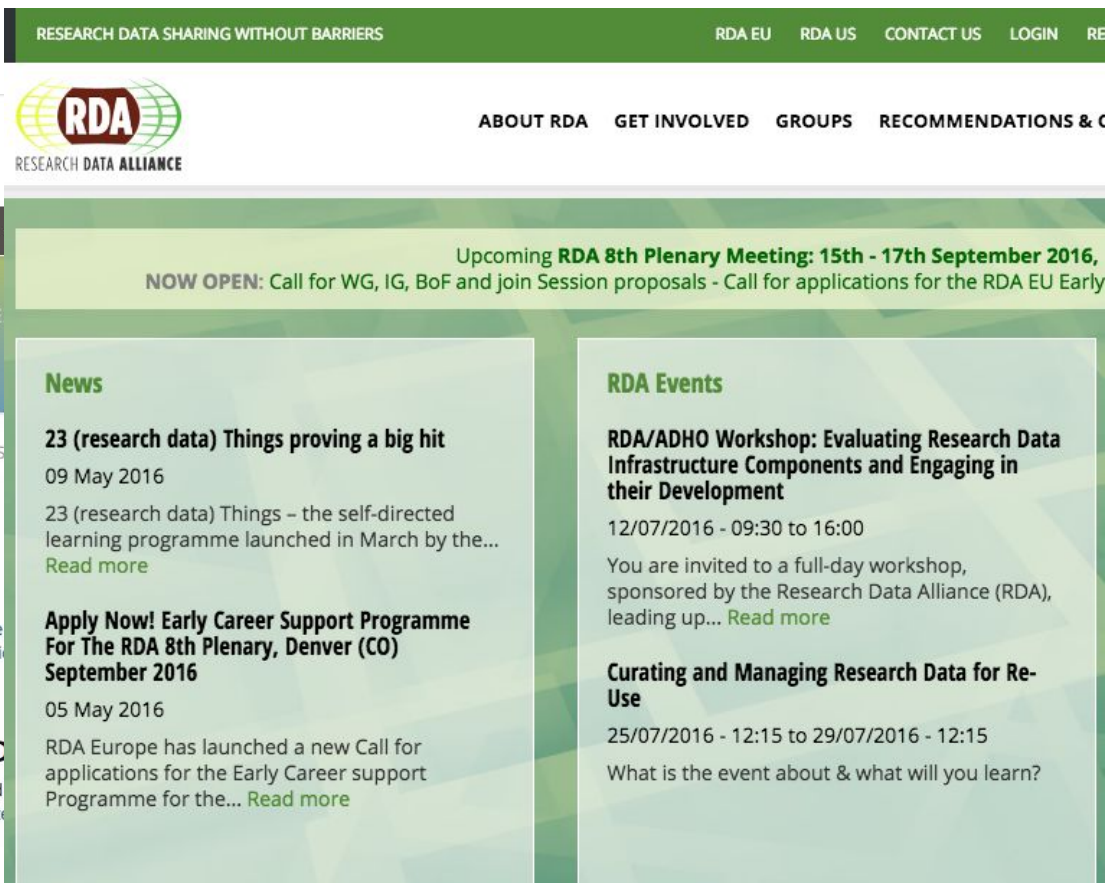
Most Popular

1 First ever conference of Research Software Engineers - call for participation – The RSE Conference (15-16 September 2016, Manchester) is...

2 Senior Research Software Engineer post at University of Manchester – The eScience Lab at the University of Manchester is...

3 Sign up for our weekly news and blog digest – #mc_embed_signup{background:#fff; clear:left;...

4 About – A national facility for cultivating and improving...

5 People – Director &...

Where some of this stuff is discussed