1. Zotero connector browser plugin:
   1. Fetch reference from github repository
      1. From bibtex file
      2. From .cff file
      3. From zenodo/doi badge on github
2. Explore crossref and eventdata
   1. Crossref DOI -> metadata: <https://search.crossref.org/>
   2. Datacite DOI -> metadata: <https://search.datacite.org/>
      1. How is this different from altmetrics?
         1. Open source, not mystery algorithm
3. Other ways people publish software (can we fetch reference information from them)
   1. Journal of Open Source Software
   2. Journal of Open Research Software
   3. SoftwareX
   4. GigaScience
4. Generate CITATION.cff based on GH info
   1. what can we get from API

Notes:

Datacite v4.2 scheme: <https://schema.datacite.org/meta/kernel-4.2/doc/DataCite-MetadataKernel_v4.2.pdf>

<https://blog.datacite.org/doi-registrations-software/>

“As of May 16, 2018, 58,301 DOIs have been registered for software. We can break down this number by repository where the software source code is hosted – most DOIs for software have been registered at Zenodo.”

CERN.ZENODO - ZENODO - Research. Shared. 41346

FIGSHARE.ARS - figshare Academic Research System 4226

PURDUE.NCIB - National Cancer Institute, Bioconductor 2769

PURDUE.EZID - Purdue University 2463

OSTI.DOE - DOE Generic 736

INIST.INRA - Institut National de Recherche Agronomique 223

OCEAN.OCEAN - Code Ocean 206

CRUI.INFNCNAF - Istituto Nazionale di Fisica Nucleare. Centro Nazionale Analisi Fotogrammi 190

CDL.UCI - UC Irvine Library 120

ETHZ.DA-RD - ETHZ Data Archive - Research Data

FORCE11 Software Citation Implementation Working Group (SCIWG): <https://github.com/force11/force11-sciwg>

* Challenges Preprint: <https://arxiv.org/abs/1905.08674v1> (or Github <https://github.com/force11/force11-sciwg/tree/master/Challenges>)

RDA software source code identification WG: <https://www.rd-alliance.org/groups/software-source-code-identification>

Browser plugin:

<https://github.com/zotero/zotero-connectors>

# Github API

[REST API endpoints](https://developer.github.com/v3/apps/available-endpoints/)

users/

* name
* company
* location

repos/

* name
* full\_name : [org]/[reponame]
* Created\_at

# Github - CFF mapping

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CFF Key | CFF Data Type | Description | Github | Github query |
| abbreviation | String | The abbreviation of the work | - | - |
| abstract | String | The abstract of a work | Repo description | <https://api.github.com/repos/org/repo> field is ‘description” |
| authors | Collection of entity or [person objects](#f6zppkp2l56p) | The author of a work | Contributor names? | <https://api.github.com/repos/org/repo>  ‘contributors\_url’ |
| collection-doi | String | The DOI of a collection containing the work |  |  |
| collection-title | String | The title of a collection or proceedings |  |  |
| collection-type | String | The type of a collection |  |  |
| commit | String | The (e.g., Git) commit hash or (e.g., Subversion) revision number of the work | Git commit hash |  |
| conference | Entity object | The conference where the work was presented |  |  |
| contact | Collection of entity or person objects | The contact person, group, company, etc. for a work |  |  |
| copyright | String | The copyright information pertaining to the work |  |  |
| data-type | String | The data type of a data set |  |  |
| database | String | The name of the database where a work was accessed/is stored |  |  |
| database-provider | Entity object | The provider of the database where a work was accessed/is stored |  |  |
| date-accessed | Date | The date the work has been last accessed |  |  |
| date-downloaded | Date | The date the work has been downloaded |  |  |
| date-published | Date | The date the work has been published |  |  |
| date-released | Date | The date the work has been released |  |  |
| department | String | The department where a work has been produced |  |  |
| doi | String | The DOI of the work |  |  |
| edition | String | The edition of the work |  |  |
| editors | Collection of entity or person objects | The editors of a work |  |  |
| editors-series | Collection of entity or person objects | The editors of a series in which a work has been published |  |  |
| end | Integer | The end page of the work |  |  |
| entry | String | An entry in the collection that constitutes the work |  |  |
| filename | String | The name of the electronic file containing the work |  |  |
| format | String | The format in which a work is represented |  |  |
| institution | Entity object | The institution where a work has been produced or published |  |  |
| isbn | String | The ISBN of the work |  |  |
| issn | String | The ISSN of the work |  |  |
| issue | Integer | The issue of a periodical in which a work appeared |  |  |
| issue-date | String | The publication date of the issue of a periodical in which a work appeared - see note below |  |  |
| issue-title | String | The name of the issue of a periodical in which the work appeared |  |  |
| journal | String | The name of the journal/magazine/newspaper/periodical where the work was published |  |  |
| keywords | Collection of strings | Keywords pertaining to the work |  |  |
| languages | Collection of ISO 639 language strings | The language of the work |  |  |
| license | License string | The license under which a work is licensed | License |  |
| license-url | String (URL) | The URL of the license text under which a work is licensed | License-URL? |  |
| location | Entity object | The location of the work |  |  |
| loc-start | Integer | The line of code in the file where the work starts |  |  |
| loc-end | Integer | The line of code in the file where the work ends |  |  |
| medium | String | The medium of the work |  |  |
| month | Integer | The month in which a work has been published |  |  |
| nihmsid | String | The NIHMSID of a work |  |  |
| notes | String | Notes pertaining to the work |  |  |
| number | String | The accession number for a work |  |  |
| number-volumes | Integer | The number of volumes making up the collection in which the work has been published |  |  |
| pages | Integer | The number of pages of the work |  |  |
| patent-states | Collection of strings | The states for which a patent is granted |  |  |
| pmcid | String | The PMCID of a work |  |  |
| publisher | Entity object | The publisher who has published the work |  |  |
| recipients | Collection of entity or person objects | The recipient of a personal communication |  |  |
| repository | String (URL) | The repository where the work is stored | Repo URL |  |
| repository-code | String (URL) | The version control system where the source code of the work is stored | Github? |  |
| repository-artifact | String (URL) | The repository where the (executable/binary) artifact of the work is stored |  |  |
| scope | String | The scope of the reference, e.g., the section of the work it adheres to |  |  |
| section | String | The section of a work that is referenced |  |  |
| senders | Collection of entity or person objects | The sender of a personal communication |  |  |
| status | Status string | The publication status of the work |  |  |
| start | Integer | The start page of the work |  |  |
| term | String | The term being referenced if the work is a dictionary or encyclopedia |  |  |
| thesis-type | String | The type of the thesis that is the work |  |  |
| title | String | The title of the work | Repo name |  |
| translators | Collection of entity or person objects | The translator of a work |  |  |
| type | Reference types string | The type of the work |  |  |
| url | String (URL) | The URL of the work | Repo URL |  |
| version | String | The version of the work | release/version? |  |
| volume | Integer | The volume of the periodical in which a work appeared |  |  |
| volume-title | String | The title of the volume in which the work appeared |  |  |
| year | Integer | The year in which a work has been published | created\_at -> extract year? |  |
| year-original | Integer | The year of the original publication | created\_at -> extract year? |  |

Person object

|  |  |  |  |
| --- | --- | --- | --- |
| CFF keys | CFF property | Github | Github query |
| family-names | String |  |  |
| given-names | String |  |  |
| name-particle | String |  |  |
| name-suffix | String |  |  |
| affiliation | String | User company | https://api.github.com/users/yochannah  Field is “company” |
| address | String |  |  |
| city | String | User location | <https://api.github.com/users/yochannah>  Field is “location” |
| region | String |  |  |
| post-code | String |  |  |
| country | String |  |  |
| orcid | String (ORCID URL) |  |  |
| email | String | User email | <https://api.github.com/users/yochannah>  Field is “email” |
| tel | String |  |  |
| fax | String |  |  |
| website | String (URL) | User blog | <https://api.github.com/users/yochannah>  Field is “blogl” |

# Citation.js

* <https://citation.js.org/demo> Try out: Can generate various citation formats from a DOI or another ID or citation format.
* <https://citation.js.org> Promises also CFF, but for now it can’t be found it in the code or the docs.
* Any new input formats with parsers can be added as plugins, and built-in ones can be overridden 
* <https://doi.org/10.7287/peerj.preprints.27466v2> (Lars G. Willighagen 2019, PeerJ Preprints)
* <https://doi.org/10.5281/zenodo.1005176> (Willighagen & Willighagen et al. 2017-2018, Zenodo)
* Issue (just like Zotero): It DOES NOT retrieve the version of software from Zenodo :-(
* Open question: Is it possible to plug in also new serialisations == output formats? That could solve the above issue

Parsing names from GitHub

https://nameparser.readthedocs.io/en/latest/

Future work - at eLife sprint:

* Deprecate node app and consider converting to a flask app, because there are helper apps
  + Use doi2cff - works ok
  + <https://github.com/citation-file-format/github2cff> doesn’t install on python 2 or 3 machines. Maybe pick it up and fix it.
  + Don’t think Citation.js is needed– unless someone wants to build this in js
  + CFF TO ZENODO?
    - Extend zenodo to pull good cff data, make the artifact deposit, and then zenodo says
      * “we suggest you update your cff to include this DOI - download the new file here
        + Could also contain your orcid if not present.
      * or copy/paste your creds into the file
      * Auto-add to cff ON github. Seems a bit scary.

# eLifeSprint

Working group: Jen Harrow ([jen.harrow@elixir-europe.org](mailto:jen.harrow@elixir-europe.org) @JenHarrow), Mateusz Kuzak ([mateusz.kuzak@gmail.com](mailto:mateusz.kuzak@gmail.com) gh: mkuzak), Sarala Wimalaratne ([sarala@datacite.org](mailto:sarala@datacite.org); @sarala), Melissa Harrison, Emmy Tsang, Leonie Mueck, Daniel Nüst, Markus Konkol, Sarthak Sehgal ([sarthaksehgal00@gmail.com](mailto:sarthaksehgal00@gmail.com), @\_sarthaksehgal)

**Problem statement:** As a life science research software developer , I would like to submit a software citation using existing open software tools to a publication, in order to make my software citable and trackable to life science funders and reusable by other life science researchers

**One-liner:** The aim is to explore the different missing pieces that are necessary to monitor software citation using existing open science tools (such as Zenodo, Zotero and GitHub), through a targeted sprint.

**Links/notes:**

[*https://www.force11.org/software-citation-principles*](https://www.force11.org/software-citation-principles)*,* [*https://www.force11.org/group/software-citation-implementation-working-group*](https://www.force11.org/group/software-citation-implementation-working-group)

[Citation File Format](https://citation-file-format.github.io) and [tools](https://github.com/citation-file-format)

[notes from BOSC CoFest](https://docs.google.com/document/d/1zza21gSszRKE82M__wsKq65R4GKGPvnNrndyYP3jdAk/edit)

[GItHub repository](https://github.com/SoftDev4Research/software-citation)

[Software Citation Checklist from Force 11](https://docs.google.com/document/d/15IiHljWa7Bf55FSdb4reNiIhFmZ7Ai_FmSslLB8o6ig/edit?pli=1#heading=h.8kw5r2j274k2)

**Resources that may be helpful:**[*https://zenodo.org/*](https://zenodo.org/)*,* [*https://www.zotero.org/*](https://www.zotero.org/)

[*https://github.com/force11/force11-sciwg*](https://github.com/force11/force11-sciwg)

## PLOS

Talking to Leonie Muck (PLOS), obut software citation in review and editorial

[Work at plos at PLOS](https://blogs.plos.org/everyone/2018/04/18/stop-hiding-your-code/)

[PLOS guideline](https://journals.plos.org/plosone/s/submission-guidelines#loc-methods-software-databases-and-tools)

## CODECHECK

<https://codecheck.org.uk>

## Challenges

Trying to loop into UI/UX st

Make user stories

Is it worth finding github collaborators to facilitate citation although done via zenodo

Steps for improving software citation

Software Developer perspective:

1. Deposit software code in git-hub
2. Assign an open source licence
3. Create a release
4. Deposit in Zenodo and get a DOI
5. Describe how to cite your software
   1. Ideally in .cff format
   2. Other options (CODEMETA, schema.org, bibtex)

Researcher / software user perspective:

1. Cite software that you used
2. Use a DOI pointing to specific version of the software within zenodo or registry eg bio.tools (but bio.tools does not use dois)

Publishers perspective :

1. Be able to include software citation easily in standard format to appear in citations list at end of publication (enables it to be picked up in google scholar etc)
2. Need DOI for check in [crossref](https://www.crossref.org/)

Funders perspective

1. Enable impact of research software to be measured by citation easier

How do we incentivise developers?

1. Knowledge on how much the software is used
2. Increased visibility of the software
3. Citation has potential to be recorded in Google Scholar
4. Use software citation numbers when applying for funding
5. Discuss with funders eg Wellcome Trust to mandate software citation for funding

Zenodo incentive to produce CFF file for software submission

ORCID in the .cff is a problem, orcids need to be validated (via single sign)

<https://esip.figshare.com/articles/Software_and_Services_Citation_Guidelines_and_Examples/7640426>

Mandatory fields

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| generic | .cff | datacite | .zenodo.json | JATS |
| Citation type |  |  |  | <ref>  <element-citation publication-type="software"> |
| Author wrapper |  | <contributors> |  | <person-group> |
| Author name wrapper |  | <contributor> |  | <name> |
| Author(s) name as string | authors:  - family-names: Druskat  given-names: Stephan | <contributorName> | creators | <string-name> |
| Author name, structured |  | <givenName> <familyName> |  | <surname>  <given-names> |
| Publication Year | date-released | <publicationYear> | created | <year> |
| Title wrapper |  | <titles> |  |  |
| Title | title | <title> | title | <data-title>  But subject to change only other current option is <article-title> |
| Version No | version | <version> | version | <version>  With optional @attribute value, eg:  <version designator="XXX"> |
| Publisher (host) | - | <publisher> | - | <publisher-name>  Could also add <publisher-loc>  Or use <source> instead to indicate repo |
| PID URL |  | Identifier |  |  |
| DOI | doi | <identifier identifierType="DOI"> | doi | <pub-id pub-id-type="doi"> |
|  |  | <ResourceType resourceTypeGeneral="Software"> |  |  |
| URL (non-PID) |  |  |  | Eg: <ext-link ext-link-type="uri" xlink:href="https://www.globalphasing.com/buster/">https://www.globalphasing.com/buster/</ext-link> |

Bibtec example out put from Zenodo:

@misc{baum\_katharina\_2019\_3363060,

author = {Baum, Katharina and

Rajapakse, Jagath C. and

Azuaje, Francisco},

title = {{Analysis of correlation-based biomolecular

networks from different omics data by fitting

stochastic block models}},

month = aug,

year = 2019,

doi = {10.5281/zenodo.3363060},

url = {https://doi.org/10.5281/zenodo.3363060}

Example of Code reference that would not have Zenodo DOI:

<ref id="bib39">

<element-citation publication-type="software">

<person-group person-group-type="author">

<name>

<surname>Schrödinger</surname>

<given-names>LLC</given-names>

</name>

</person-group>

<year iso-8601-date="2011">2011</year>

<source>The PyMOL Molecular Graphics System</source>

<publisher-name>Schrödinger LLC</publisher-name>

<version designator="1.2.3">1.2r3pre</version>

<ext-link ext-link-type="uri" xlink:href='https://www.pymol.org/'>https://www.pymol.org/</ext-link>

</element-citation>

Also citing R:

R Core Team (2013). R: A language and environment for statistical

computing. R Foundation for Statistical Computing, Vienna, Austria.

URL http://www.R-project.org/.

CodeMeta work - metadata fields for software citation

<https://codemeta.github.io/>

<https://raw.githubusercontent.com/codemeta/codemeta/2.0/codemeta.jsonld>

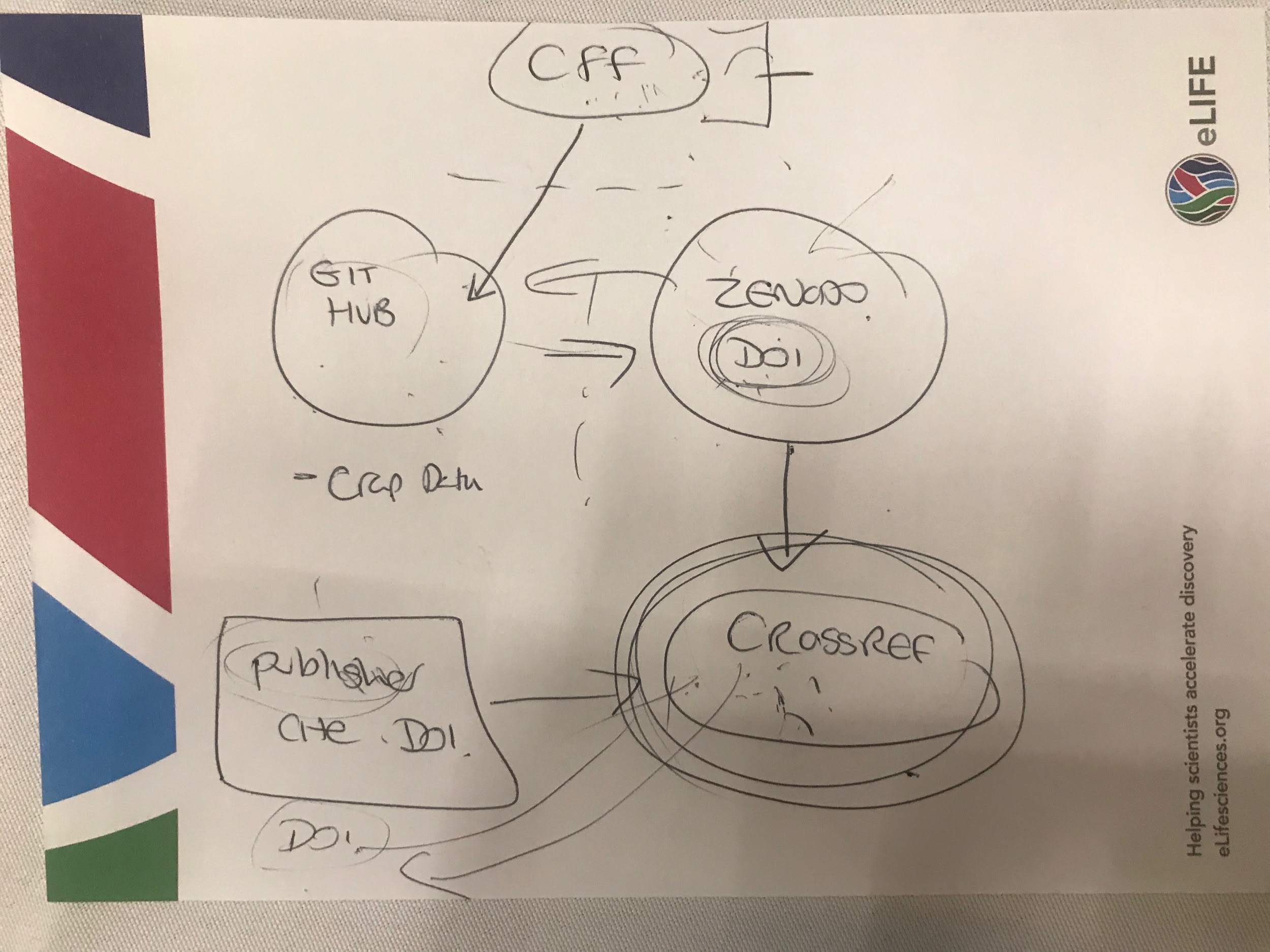
Make your code citable using GitHub and Zenodo - <https://genr.eu/wp/cite/>

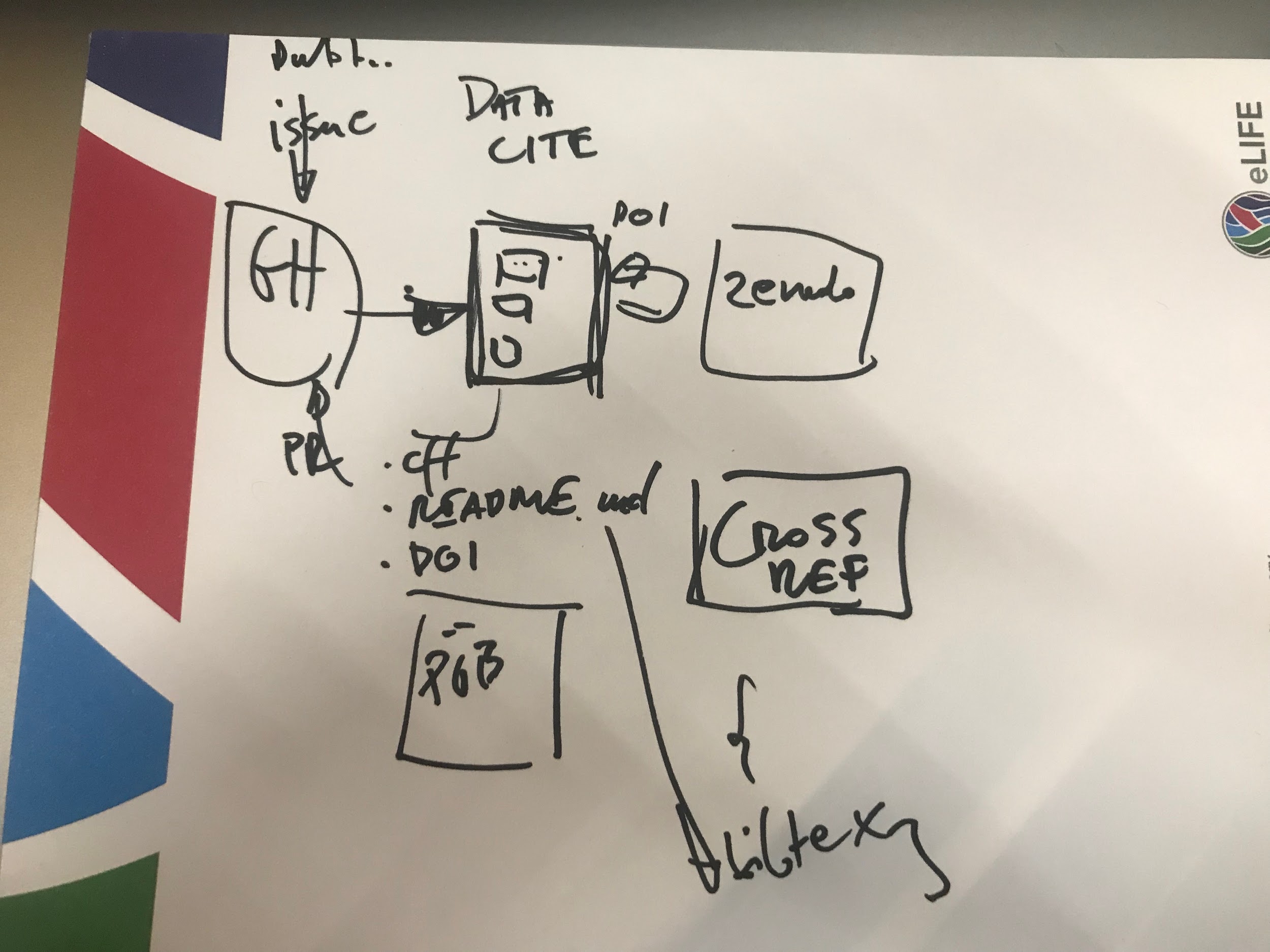
What is needed for this user journey:

Possibility 1:

* Software to produce CFF file with metadata and allow editing of file to change (authors , name etc ) (not developed yet) would run on zenodo as part of `DOI registration’
  + Need zenodo collaboration pre-register DOI and template
* CFF file is pulled from zenodo to github repo

Possibility 2:





GitHub API examples:

<https://github.com/marketplace/dependabot-preview>

<https://github.com/elifesciences/annotations/pull/271>

<https://developer.github.com/v3/>

## Architecture Design

backend

Simple form frontend

Talk to GitHub via the API

Talk to Zenodo via the API

# Plan for Day 2

* Look into [Pyrdm](https://github.com/pyrdm/pyrdm) for pre-registration of DOI if applicable
* We have virtual VM working
* Follow up on architectural design
* Look at Datacite incorporation

[Slides from time to shine](https://docs.google.com/presentation/d/1dGERl3b68Crsnc2wptFjzTNkgmcYNUs6kyUqeZN48GE/edit?usp=sharing)

The PoC application