Software Metadata Extraction from Code Repositories: An overview

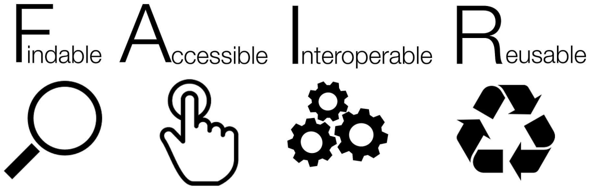
**Daniel Garijo, Ontology Engineering Group,
Universidad Politécnica de Madrid, Spain**

✉ daniel.garijo@upm.es
@dgarijov

Software Metadata

Now you are familiar with **Codemeta**

- Community standard extending Schema.org
- **JSON-LD**
- Metadata is key for **FAIR**



https://codemeta.github.io/terms/

Metadata is key for Research Software Understanding

Describe

Given a software project:

- What is it about?
- Examples?
- Relation to other resources (data, papers)?
- Metadata?

Compare

Given two or more tools:

- What are their similarities?
- Differences?
- Main features?

Reuse

How to quickly:

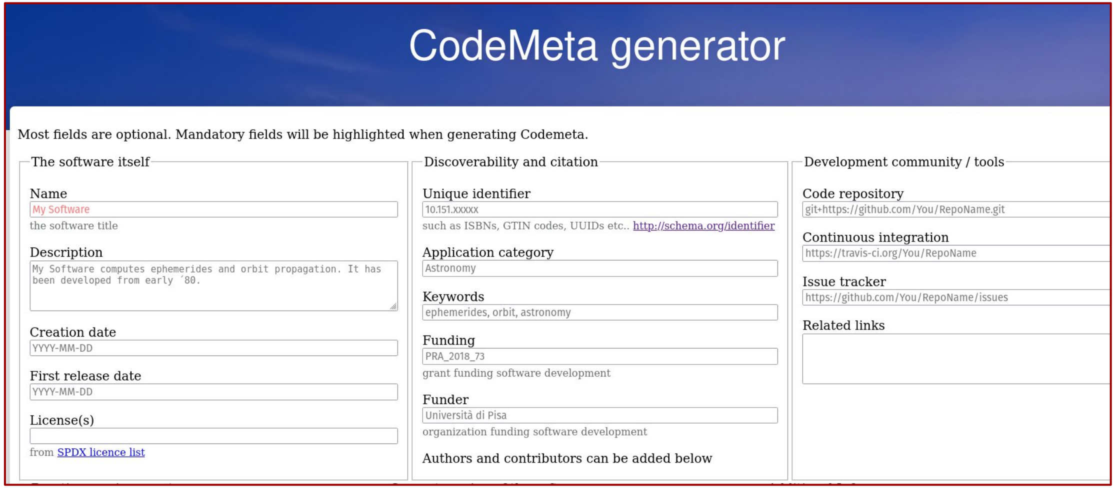
- run?
- repeat?
- reproduce?
- fix?
- combine?

Research Software metadata Extraction

Manually

- Codemeta generator

  https://codemeta.github.io/codemeta-generator/



CodeMeta generator

Most fields are optional. Mandatory fields will be highlighted when generating Codemeta.

The software itself

Name
My Software
the software title

Description
My Software computes ephemerides and orbit propagation. It has been developed from early ´80.

Creation date
YYYY-MM-DD

First release date
YYYY-MM-DD

License(s)

from SPDX licence list

Discoverability and citation

Unique identifier
10.151.xxxxx
such as ISBNs, GTIN codes, UUIDs etc.. http://schema.org/identifier

Application category
Astronomy

Keywords
ephemerides, orbit, astronomy

Funding
PRA_2018_73
grant funding software development

Funder
Università di Pisa
organization funding software development

Authors and contributors can be added below

Development community / tools

Code repository
git+https://github.com/You/RepoName.git

Continuous integration
https://travis-ci.org/You/RepoName

Issue tracker
https://github.com/You/RepoName/issues

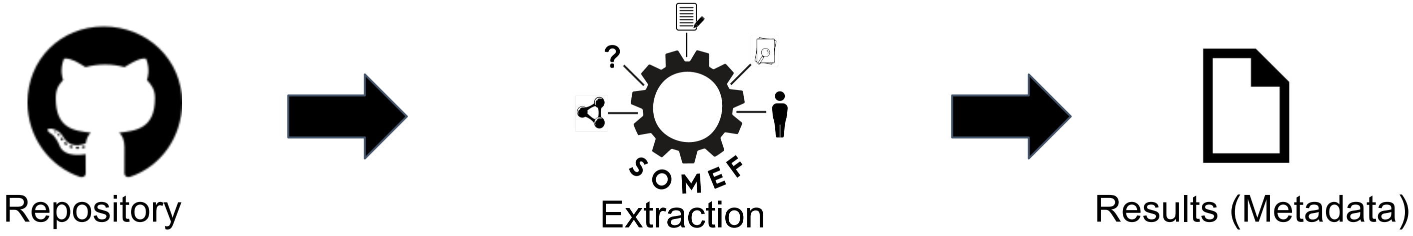
Related links

Automatically

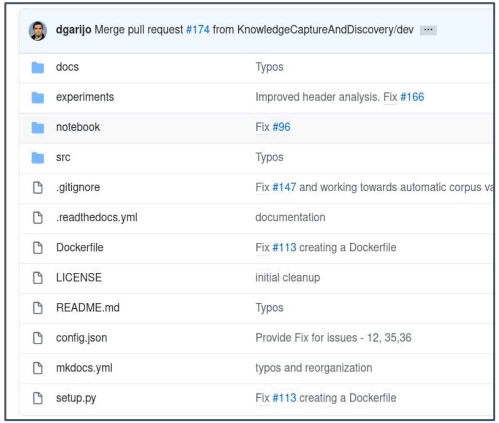
- Software Metadata Extraction Framework (SoMEF)

Text classification: Software Metadata Extraction Framework

https://github.com/KnowledgeCaptureAndDiscovery/somef/



Repository → Extraction → Results (Metadata)



dgarijo Merge pull request #174 from KnowledgeCaptureAndDiscovery/dev ···

docs	Typos
experiments	Improved header analysis. Fix #166
notebook	Fix #96
src	Typos
.gitignore	Fix #147 and working towards automatic corpus va
.readthedocs.yml	documentation
Dockerfile	Fix #113 creating a Dockerfile
LICENSE	initial cleanup
README.md	Typos
config.json	Provide Fix for issues - 12, 35,36
mkdocs.yml	typos and reorganization
setup.py	Fix #113 creating a Dockerfile

- **Readme Analysis**
  - Supervised classification
  - Regular expressions
  - Header analysis
- **File exploration**
  - Notebooks
  - Dockerfiles
  - Documentation
- **GitHub API**

CodeMeta

TURTLE

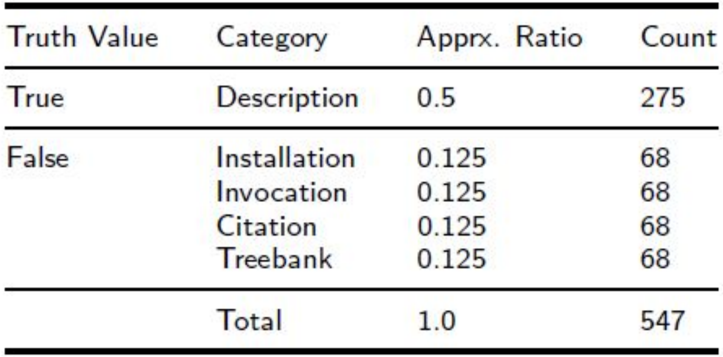
JSON

+ provenance

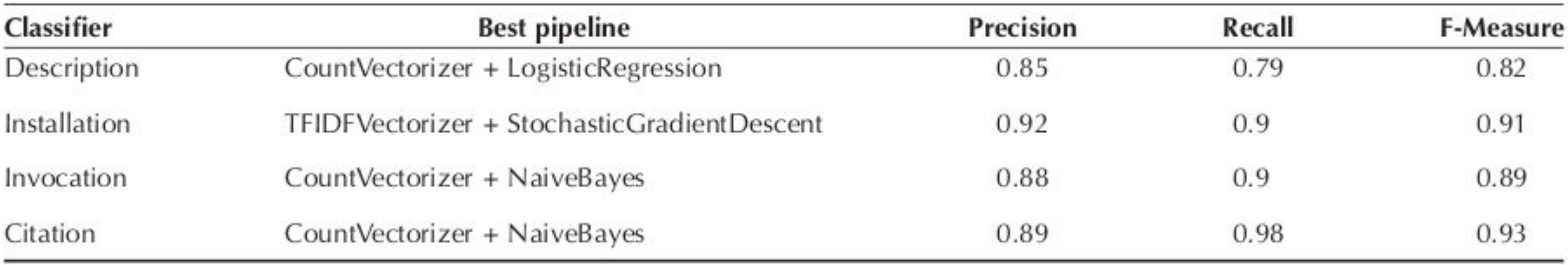
Kelley, A., & Garijo, D. (2021). A framework for creating knowledge graphs of scientific software metadata. *Quantitative Science Studies*, 1-37.

SOMEF: Supervised classification

- Paragraph-based **text classification**
- Four main categories (binary classification):
  - Installation
  - Description
  - Invocation



Truth Value	Category	Apprx. Ratio	Count
True	Description	0.5	275
False	Installation	0.125	68
	Invocation	0.125	68
	Citation	0.125	68
	Treebank	0.125	68
	Total	1.0	547

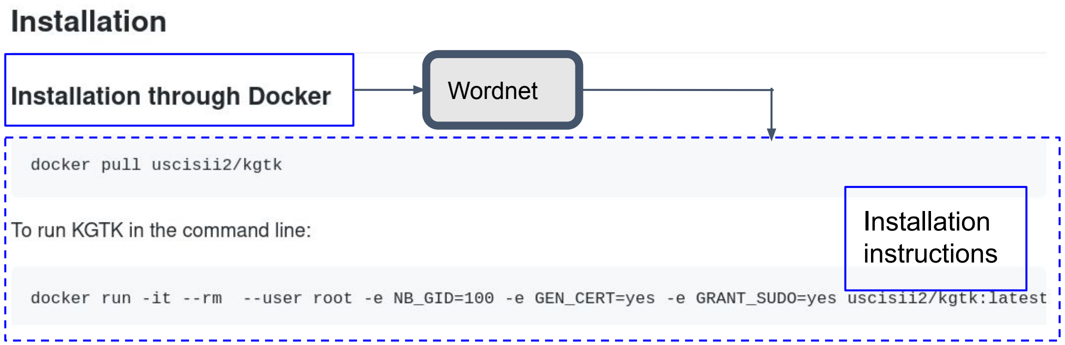


Classifier	Best pipeline	Precision	Recall	F-Measure
Description	CountVectorizer + LogisticRegression	0.85	0.79	0.82
Installation	TFIDFVectorizer + StochasticGradientDescent	0.92	0.9	0.91
Invocation	CountVectorizer + NaiveBayes	0.88	0.9	0.89
Citation	CountVectorizer + NaiveBayes	0.89	0.98	0.93

Simple classification pipelines yield **adequate** results

SOMEF: File Exploration and Regular Expressions

- Extraction based on frequent header analysis
  - Fuzzy matching based on synsets



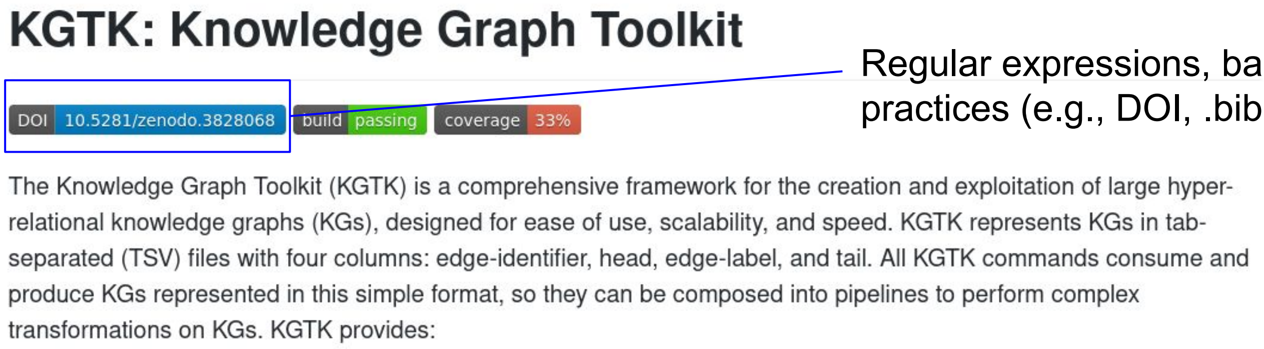
Installation

Installation through Docker

```
docker pull uscisii2/kgtk
```

To run KGTK in the command line:

```
docker run -it --rm  --user root -e NB_GID=100 -e GEN_CERT=yes -e GRANT_SUDO=yes uscisii2/kgtk:latest
```



KGTK: Knowledge Graph Toolkit

DOI 10.5281/zenodo.3828068 build passing coverage 33%

The Knowledge Graph Toolkit (KGTK) is a comprehensive framework for the creation and exploitation of large hyper-relational knowledge graphs (KGs), designed for ease of use, scalability, and speed. KGTK represents KGs in tab-separated (TSV) files with four columns: edge-identifier, head, edge-label, and tail. All KGTK commands consume and produce KGs represented in this simple format, so they can be composed into pipelines to perform complex transformations on KGs. KGTK provides:

Regular expressions, based on common practices (e.g., DOI, .bib, etc.)

SOMEF: Classifying software types based on README files

Using READMEs to **categorize** software

- Creating a methodology to recognize categories based on awesome lists

… and recognize their **nature**:

- Workflow, website, library, tool, ontology…

Readme
PYLEOCLIM

“Data Analysis”
“Package”

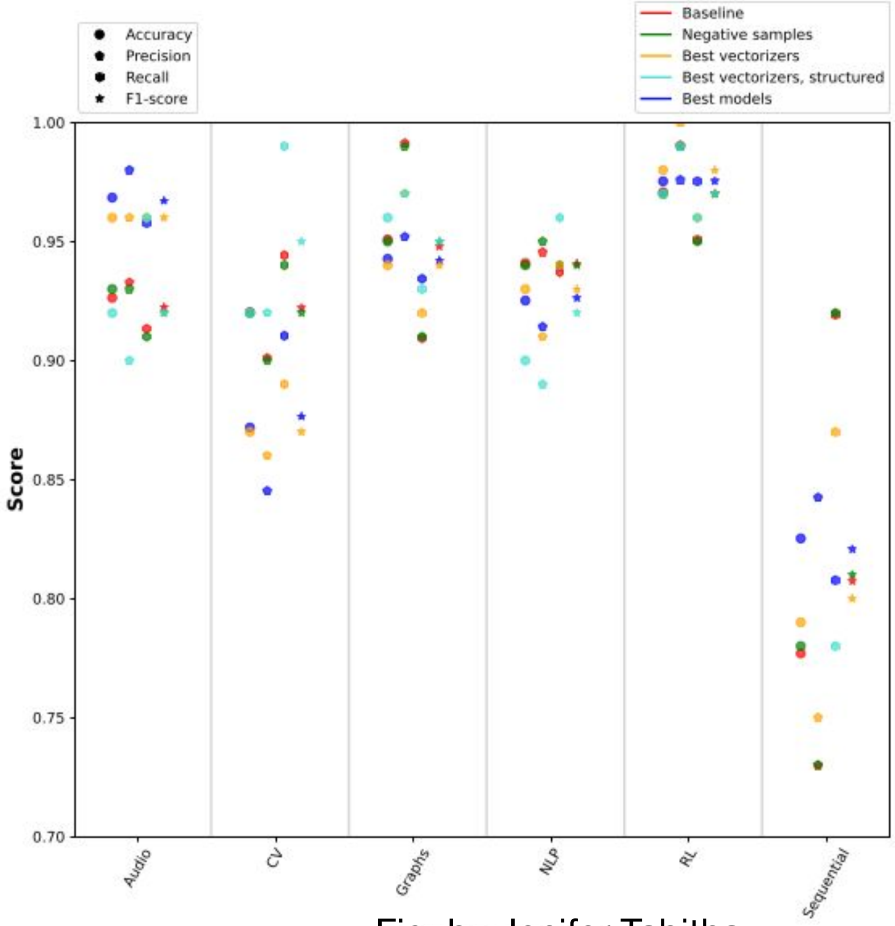


Fig. by Jenifer Tabitha

SOMEF: Recognizing Metadata Categories

- Name (GA)
- Full title (RE)
- Description (SC, HA)
- Citation (SC, RE, HA)
- Installation instructions (SC, HA)
- Invocation (SC)
- Usage examples (HA)
- Documentation (HA, FE)
- Requirements (HA)
- Contributors (HA)
- FAQ (HA)
- Support (HA)
- License (GA, HA, FE)
- Stars (GA)

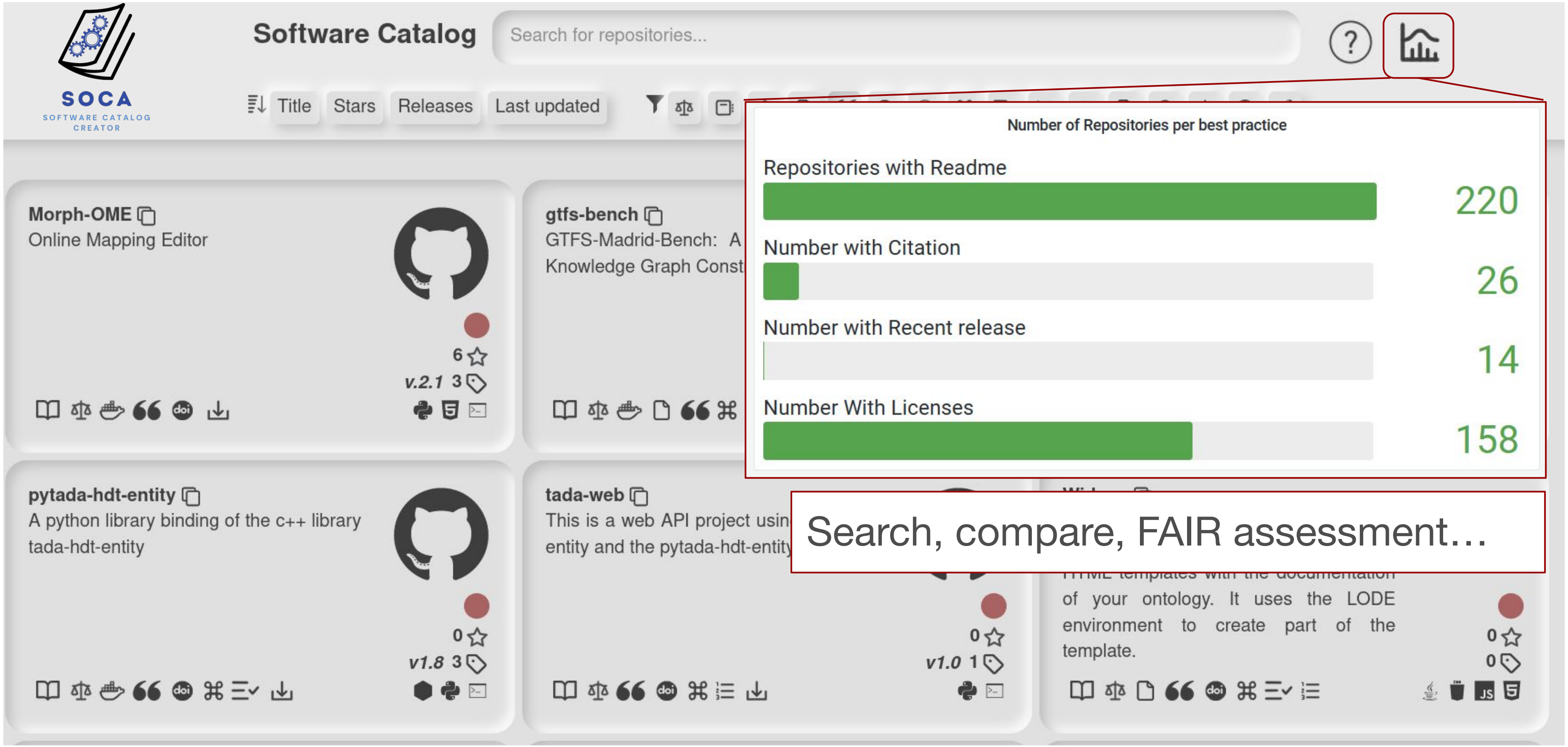
Method used (provenance):

- Supervised Classification (SC)
- Header Analysis and Synset comparison (HA)
- File Exploration (FE)
- Regular Expressions (RE)
- GitHub API (GA)

- Contact (HA)
- Download URL (HA, GA)
- DOI (RE)
- DockerFile (FE)
- Notebooks (FE)
- Executable notebooks (Binder, Collab) (RE)
- Owner: (GA)
- Keywords (GA)
- Source code (GA)
- Releases (GA)
- Changelog (GA)
- Issue tracker (GA)
- Programming languages (GA)
- Acknowledgements (HA)
- Logos (RE)
- Images (RE)
- Shell scripts (FE)
- Code of conduct (FE)
- Repository status (RE)
- Arxiv links (RE)
- Support channels (RE)
- Software category (SC)
- …

So what can you achieve once you have **rich metadata**?

Early result: Automated customized software catalogs



SOCA
SOFTWARE CATALOG CREATOR

Software Catalog

Search for repositories...

Title | Stars | Releases | Last updated

Morph-OME
Online Mapping Editor
6
v.2.1 3

gtfs-bench
GTFS-Madrid-Bench: A
Knowledge Graph Const

pytada-hdt-entity
A python library binding of the c++ library tada-hdt-entity
0
v1.8 3

tada-web
This is a web API project usin
entity and the pytada-hdt-entity
0
v1.0 1

of your ontology. It uses the LODE environment to create part of the template.
0
0

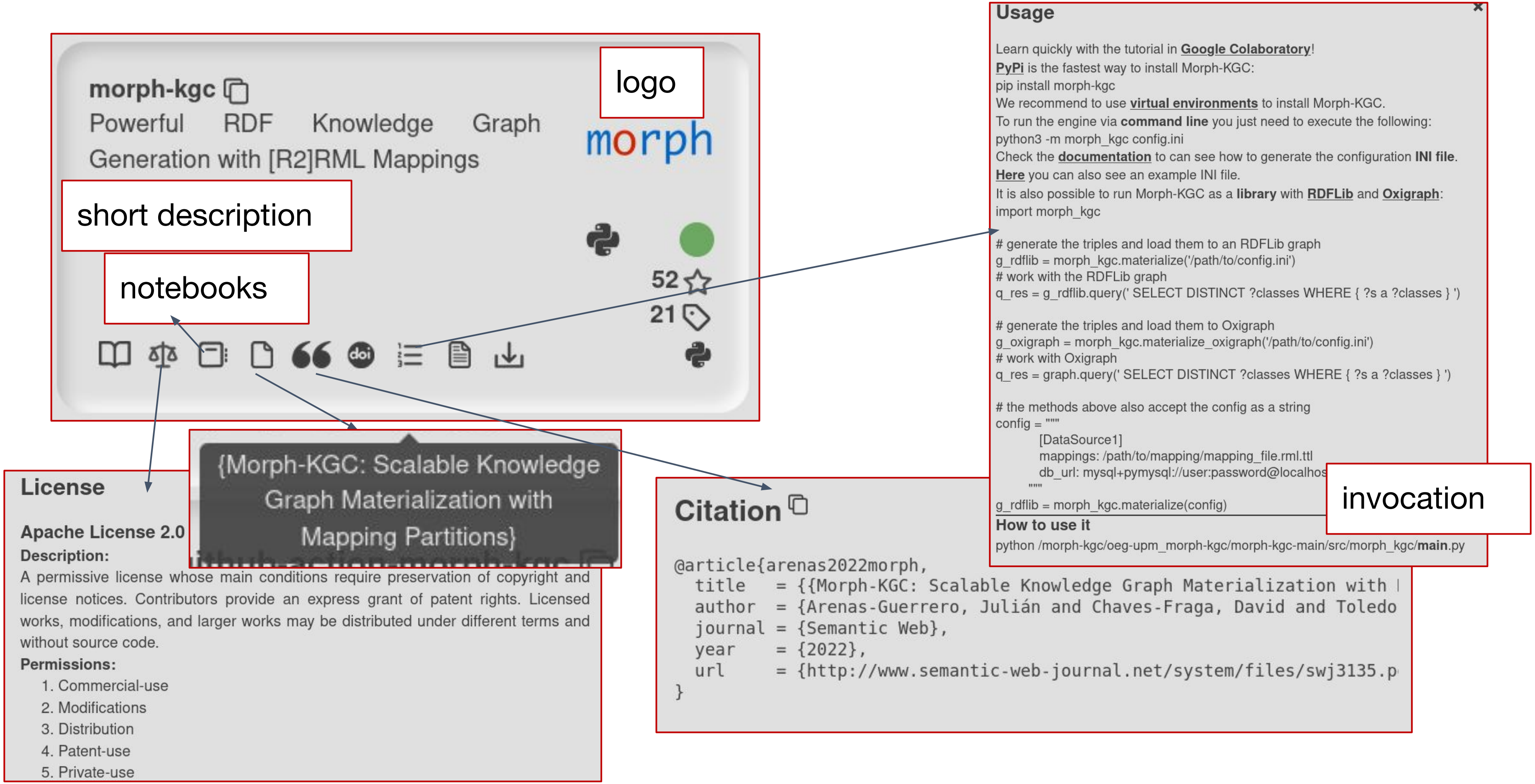
Number of Repositories per best practice

Repositories with Readme	220
Number with Citation	26
Number with Recent release	14
Number With Licenses	158

Search, compare, FAIR assessment…

Alpha available at: https://software.oeg.fi.upm.es/ Github: https://github.com/oeg-upm/soca

A software repository at a glance



morph-kgc

Powerful RDF Knowledge Graph Generation with [R2]RML Mappings

morph

52

21

{Morph-KGC: Scalable Knowledge Graph Materialization with Mapping Partitions}

ithub-action-morph-kgc

License

Apache License 2.0

Description:

A permissive license whose main conditions require preservation of copyright and license notices. Contributors provide an express grant of patent rights. Licensed works, modifications, and larger works may be distributed under different terms and without source code.

Permissions:

1. Commercial-use
2. Modifications
3. Distribution
4. Patent-use
5. Private-use

Citation

```
@article{arenas2022morph,
  title   = {{Morph-KGC: Scalable Knowledge Graph Materialization with
  author  = {Arenas-Guerrero, Julián and Chaves-Fraga, David and Toledo
  journal = {Semantic Web},
  year    = {2022},
  url     = {http://www.semantic-web-journal.net/system/files/swj3135.p
}
```

Usage

Learn quickly with the tutorial in **Google Colaboratory**!

PyPi is the fastest way to install Morph-KGC:

pip install morph-kgc

We recommend to use **virtual environments** to install Morph-KGC.

To run the engine via **command line** you just need to execute the following:

python3 -m morph_kgc config.ini

Check the **documentation** to can see how to generate the configuration **INI file**.

Here you can also see an example INI file.

It is also possible to run Morph-KGC as a **library** with **RDFLib** and **Oxigraph**:

```
import morph_kgc

# generate the triples and load them to an RDFLib graph
g_rdflib = morph_kgc.materialize('/path/to/config.ini')
# work with the RDFLib graph
q_res = g_rdflib.query(' SELECT DISTINCT ?classes WHERE { ?s a ?classes } ')

# generate the triples and load them to Oxigraph
g_oxigraph = morph_kgc.materialize_oxigraph('/path/to/config.ini')
# work with Oxigraph
q_res = graph.query(' SELECT DISTINCT ?classes WHERE { ?s a ?classes } ')

# the methods above also accept the config as a string
config = """
        [DataSource1]
        mappings: /path/to/mapping/mapping_file.rml.ttl
        db_url: mysql+pymysql://user:password@localhos
        """
g_rdflib = morph_kgc.materialize(config)
```

How to use it

python /morph-kgc/oeg-upm_morph-kgc/morph-kgc-main/src/morph_kgc/**main**.py

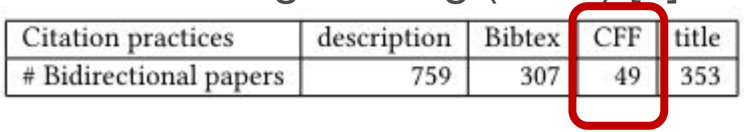
A quick reality check…

A quick reality check…

We still have a **long way ahead of us**:

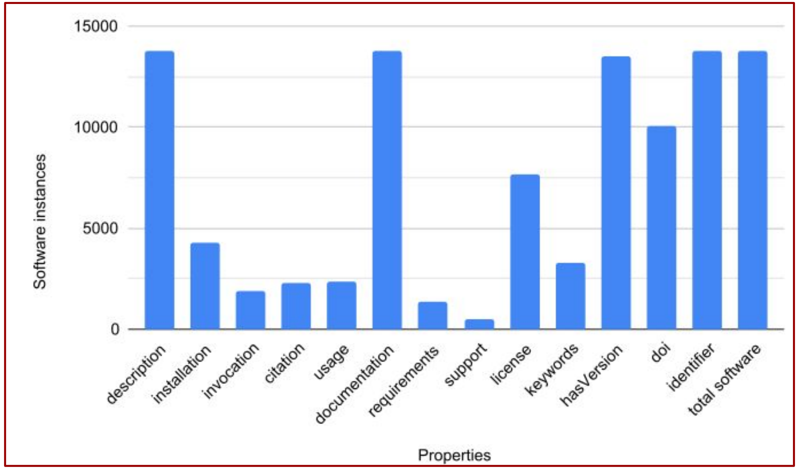
- Best practices not widely adopted
- Citation practices are heterogeneous

Software Engineering (Arxiv) [2]

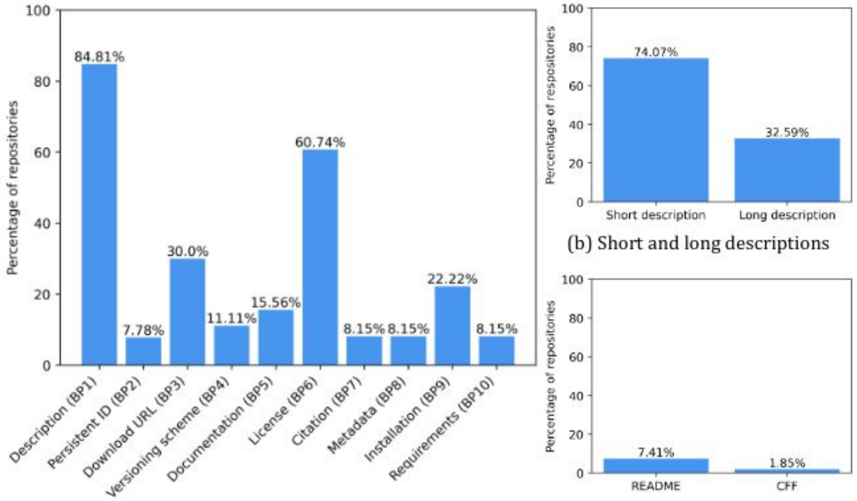


Citation practices	description	Bibtex	CFF	title
# Bidirectional papers	759	307	49	353

Zenodo [1]



My **own lab** [3] :(



(b) Short and long descriptions

[1] Kelley, A., & Garijo, D. (2021). A framework for creating knowledge graphs of scientific software metadata. *Quantitative Science Studies*, 1-37.

[2] Garijo, D.; Arroyo, M.; Gonzalez, E.; Treude, C.; and Tarocco, N. Bidirectional Paper-Repository Tracing in Software Engineering. To appear in 21st International Conference on Mining Software Repositories, Cham, 2024. ACM

[3] Iglesias-Molina, A.; and Garijo, D. Towards Assessing FAIR Research Software Best Practices in an Organization Using RDF-star. Semantics 2023 Posters and Demos (CEUR), 3526. 2023.

Conclusions

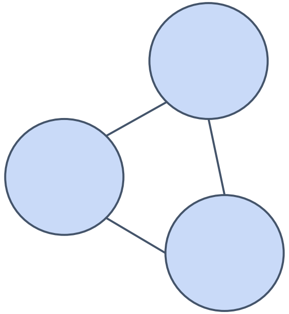
Research software **should become FAIR**!

Automated metadata extraction

- Scalable
- Helps fixing issues at the source (maintainable)
- Less effort
- But may contain **errors**

Next steps: beyond **Codemeta**

- inputs/ outputs
- models
- workflows
- containers
- How to make the extracted metadata **more actionable**?

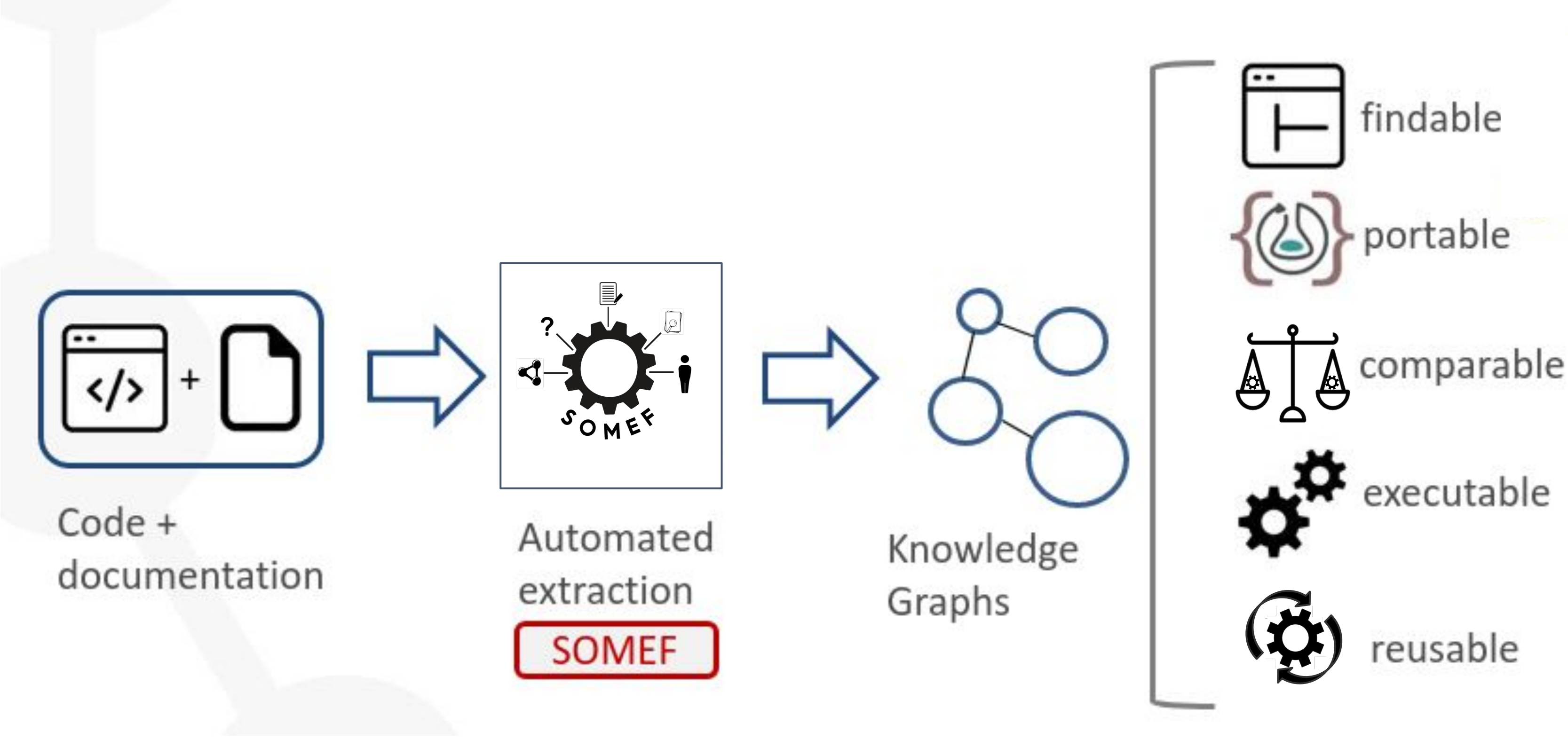


Acknowledgements

Thanks to Yolanda Gil, Varun Ratnakar, Maximiliano Osorio, Hernán Vargas, Deborah Khider, Allen Mao, Aidan Kelley, Haripriya Dharmala, Jiajing Wang, Rosa Filgueira, Pablo Calleja, Oscar Corcho, Laura Camacho, Jhon Toledo, Miguel Angel García, Esteban Gonzalez, Elena Montiel, Elvira Amador & all the students at UPM and USC who participated in the initiatives mentioned in this presentation

This work has been supported by the Madrid Government (Comunidad de Madrid-Spain) under the Multiannual Agreement with Universidad Politécnica de Madrid in the line Support for R&D projects for Beatriz Galindo researchers, in the context of the V PRICIT (Regional Programme of Research and Technological Innovation)

Questions?



Code + documentation

Automated extraction

SOMEF

Knowledge Graphs

findable

portable

comparable

executable

reusable

Let's create **machine-actionable** software metadata to promote Open Science!