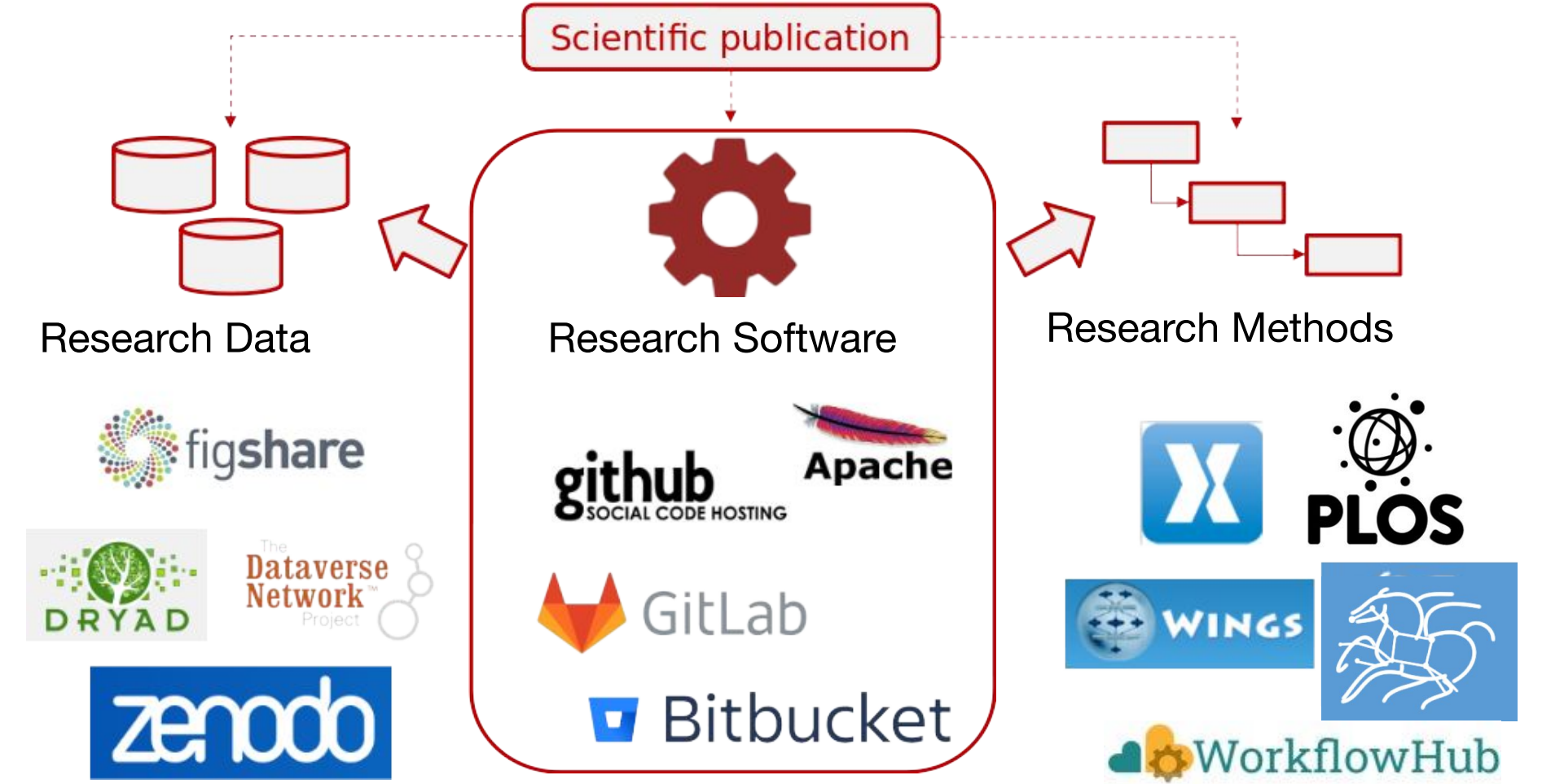
Open Science, Research Software and AI: A year at the Ontology Engineering Group

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

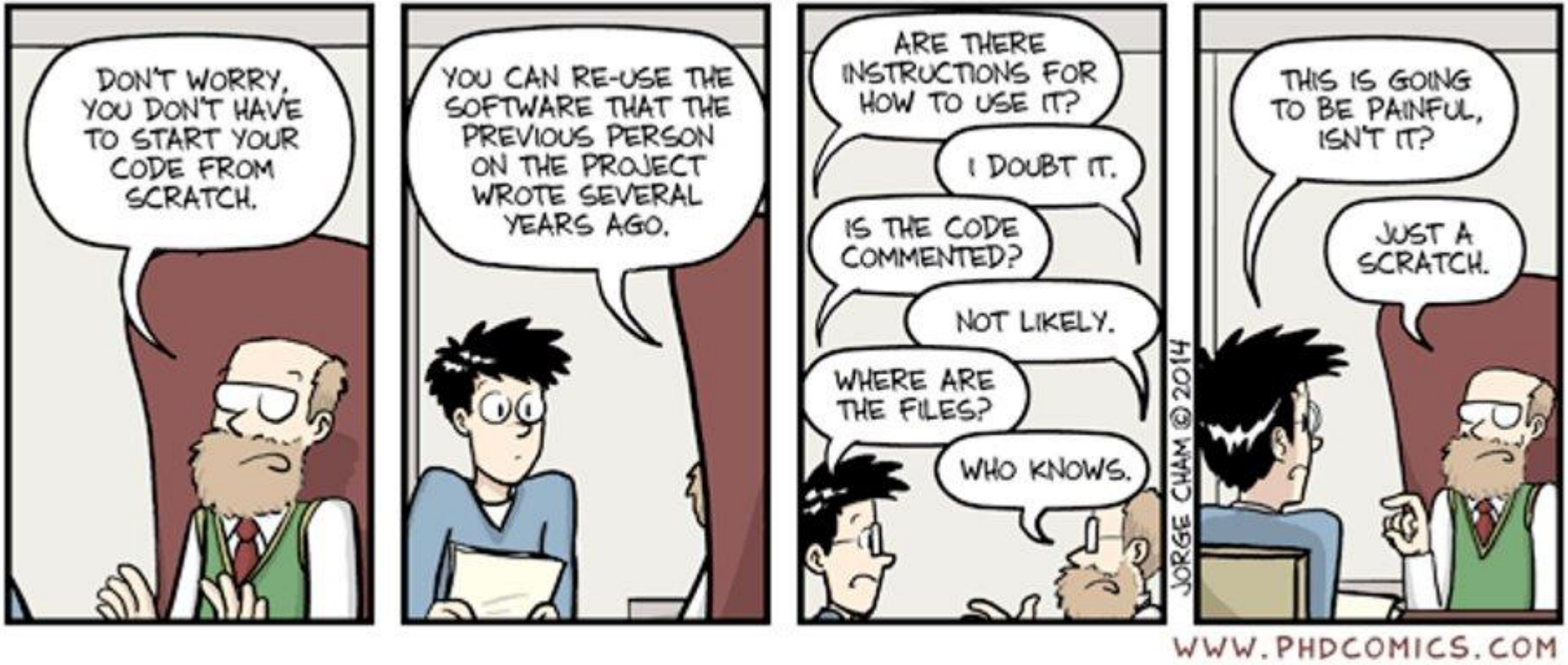
Slides based on my previous group report (July, 2021): https://docs.google.com/presentation/d/14MExu9NXPMFhwSDg2bd_uZAWfhuYsCNbmbfj1B42jQU/edit?usp=sharing

daniel.garijo@upm.es
@dgarijov

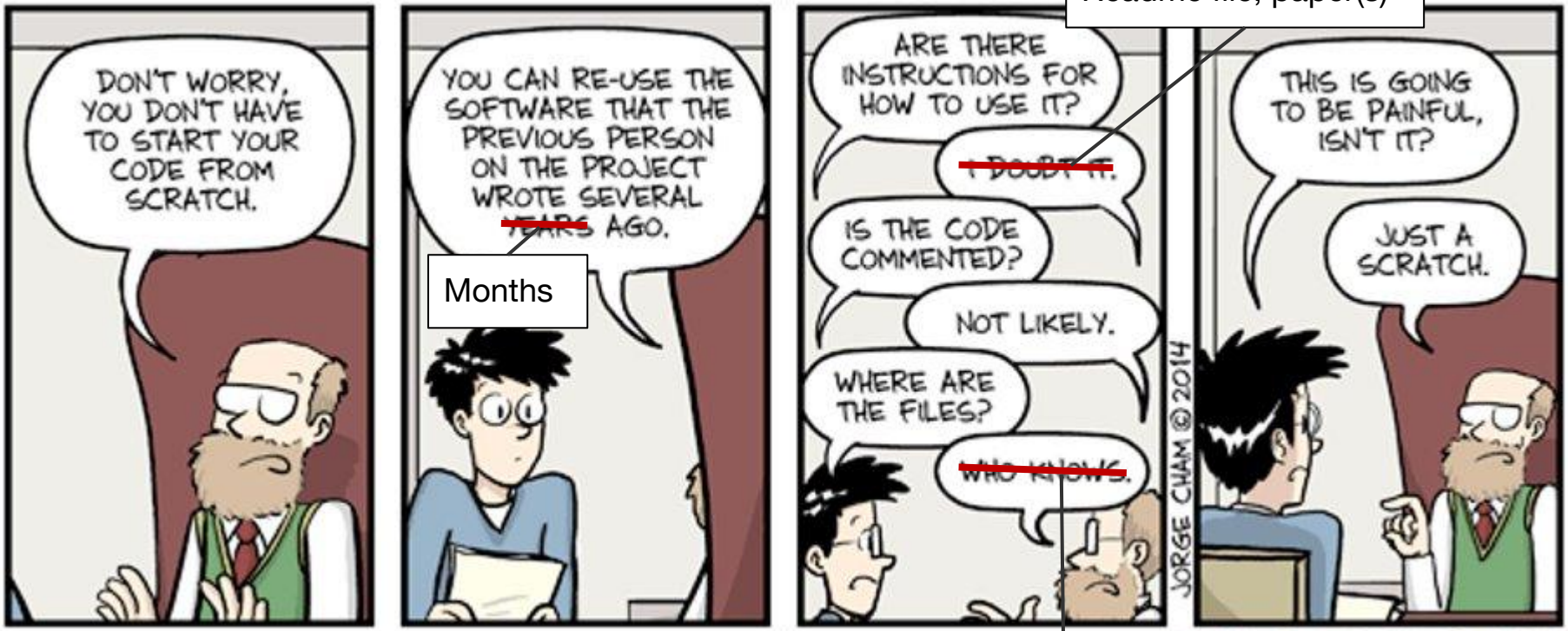
Research Software is one of the pillars of Open Science



Is this situation familiar?



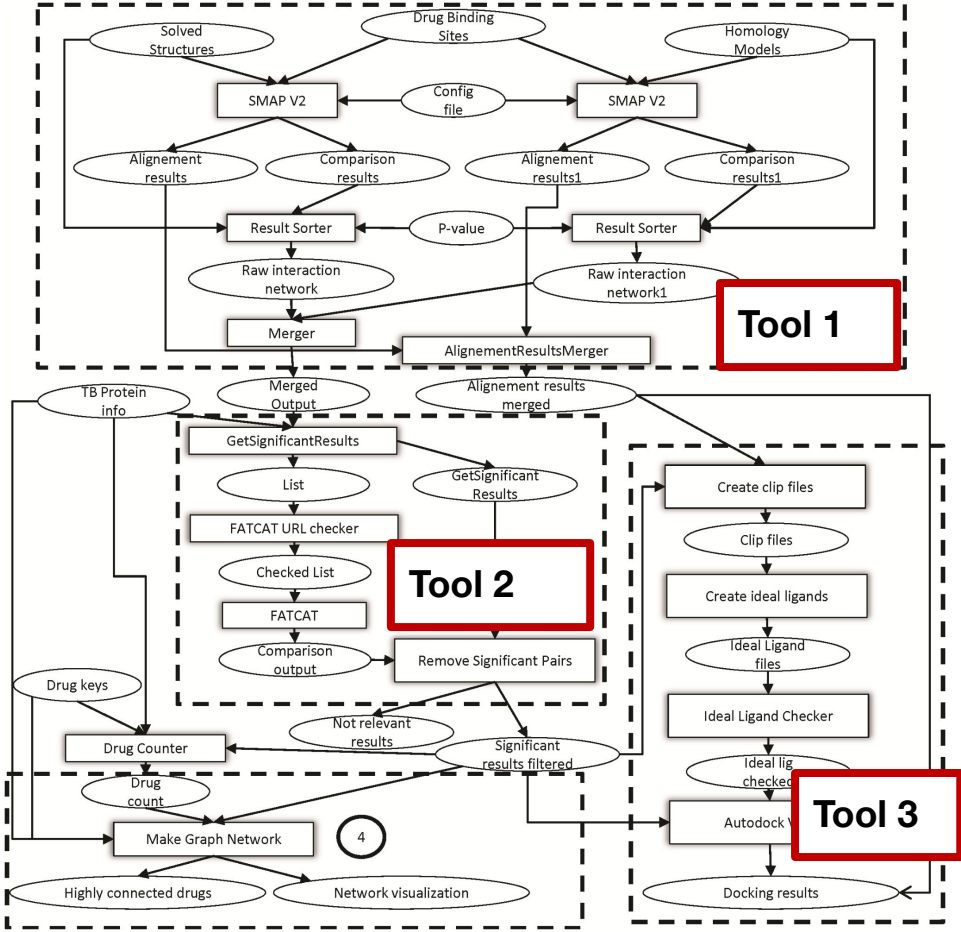
Or this one?


DON'T WORRY, YOU DON'T HAVE TO START YOUR CODE FROM SCRATCH.
YOU CAN RE-USE THE SOFTWARE THAT THE PREVIOUS PERSON ON THE PROJECT WROTE SEVERAL ~~YEARS~~ AGO.
Months
ARE THERE INSTRUCTIONS FOR HOW TO USE IT?
~~I DOUBT IT.~~
IS THE CODE COMMENTED?
NOT LIKELY.
WHERE ARE THE FILES?
~~WHO KNOWS.~~
JORGE CHAM © 2014
THIS IS GOING TO BE PAINFUL, ISN'T IT?
JUST A SCRATCH.
WWW.PHDCOMICS.COM
Readme file, paper(s)
GitHub, Gitlab, Docker image etc.

Reusability takes time, even when sources are available

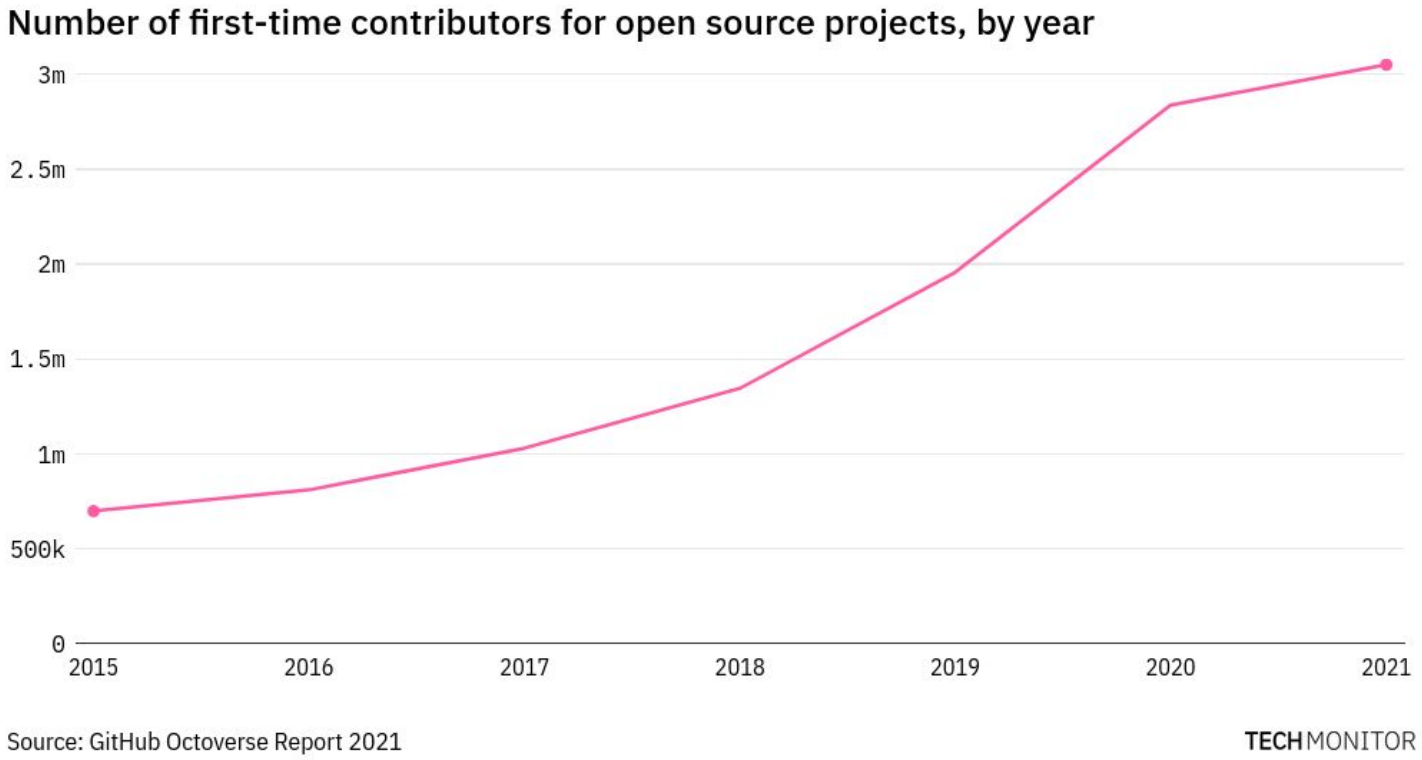
In [1] we tried to reproduce an effort from **one** year before.

- All data were available online
- All tools were available online (except one, but authors had a replacement)
- > 250 hrs to full reproducibility
- > 100 hrs to get familiar with the tools and their I/O



[1] Garijo, D., Kinnings, S., Xie, L., Xie, L., Zhang, Y., Bourne, P. E., & Gil, Y. (2013). Quantifying reproducibility in computational biology: the case of the tuberculosis drugome. *PloS one*, *8*(11), e80278.

Millions of open-source repositories are updated/created every year



Can we automatically accelerate **software understanding**?

The dimensions of Software Understanding

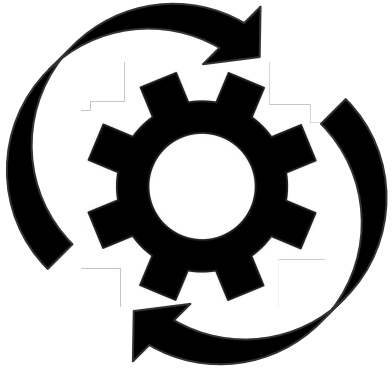
Describe

- what is in it?
- what's about?
- examples?
- relation to other resources?

Compare

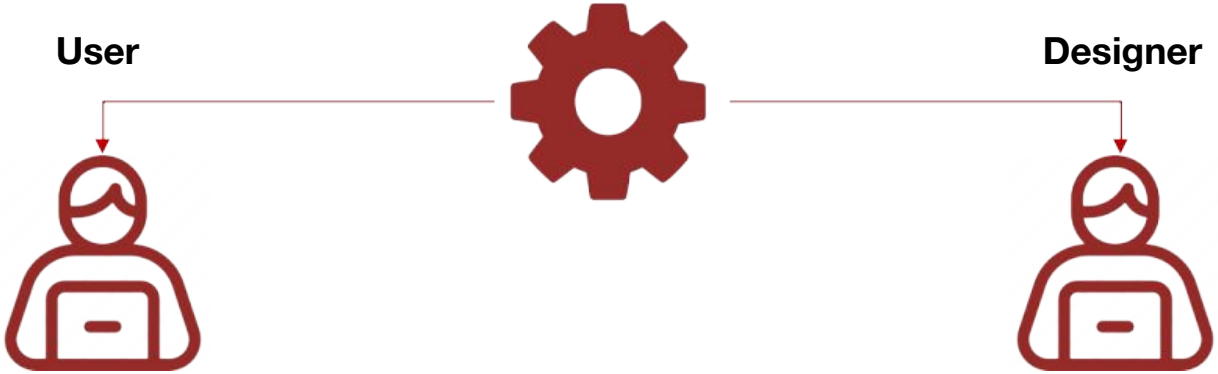
- similarities?
- differences?
- features?

Reuse



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

Challenges depend on user perspectives

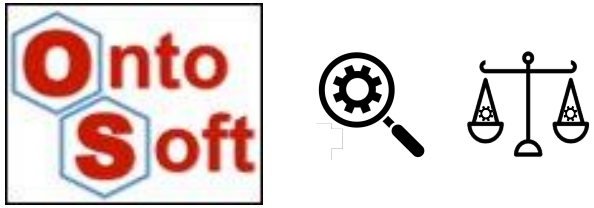


- What does the software component do? Which of its methods should I use?
- How to transform my data to use the software component?
- How to interpret the results produced by the software component?
- How to invoke the software component?
- How to configure the software component with the right parameters?
- How to compare against similar methods?

- How to ease capturing the dependencies and installation instructions of my software?
- How to encapsulate my software so it can be used with other data?
- How to describe my software so it can be used by others?
- How to test if my software is ready to be used by others?

Before returning to the Ontology Engineering Group…

OntoSoft: Creating a Software registry for the Geosciences



https://ontosoft.org/

OntoSoft: Capturing Scientific Software Metadata. Eighth ACM International Conference on Knowledge Capture, Palisades, NY, 2015

Representing Software Metadata: OntoSoft

Crowdsourced Software Metadata Registry

- Complements code repositories to make them understandable
- Software metadata designed for scientists
- Metadata is curated by decentralized communities of users
- Training scientists on best practices

http://ontosoft.org

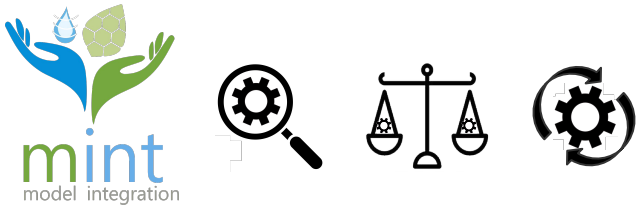
OntoSoft Software Community Training

Compare Software

DrEICH algorithm, PIHM, PIHMgis, TauDEM, WBMsed

PIHM	PIHMgis	DrEICH	TauDEM	WBMsed
				
What are domain specific keywords for this software ? (eg: hydrology, climate)				
Geomorphology, Hydrological, Bedrock channel ero-	Basins, Continental	Basins, GIS	Hydrologically corrected DEM, Watershed	Sediment flux, Global model, Hydrological model
What Operating Systems can the software run on ?				
Unix Linux	Unix Windows Linux Mac OS	Unix Windows Linux Mac OS	Unix Windows Linux Mac OS	Unix Linux
Is there any test data available for the software ?				
Test Data Location: http://onlinelibrary.wiley.-com/doi/10.1002/2013WR015167/full **Test Data Description:** Two test DEMs are included in the repository,	**Test Data Location:** http://source-forge.net/projects/pih-mmodel/ **Test Data Description:** Upper Juniata River 875 km^2: see: http://source-		**Test Data Location:** http://csdms.colorado.e-du/wiki/Model:Tau-DEM#Testing **Test Data Description:** The Logan River DEM is a small test dataset useful	**Test Data Location:** http://csdms.colorado.e-du/wiki/Mod-el:WBMsed#Testing **Test Data Description:** Extensive input dataset is available on the CSDMS

MINT: Facilitating Model Integration in the Geosciences



https://mint-project.info/

Yolanda Gil, Daniel Garijo, Deborah Khider, et.al. 2021. Artificial Intelligence for Modeling Complex Systems: Taming the Complexity of Expert Models to Improve Decision Making. *ACM Trans. Interact. Intell. Syst.* 11, 2, Article 11 (June 2021), 49 pages. DOI:https://doi.org/10.1145/3453172

Adding structure to software Metadata: MINT (Model Integration)

Low grained machine-readable Software Metadata:

- (From OntoSoft) Attribution, license, funding, usage examples...
- Executable software components
- Software invocation
- Input & output files, variables and units
- Containers used to encapsulate and run software component

IO Files:

	Name	Description
INPUT	pihm-riv	Spatial geometry and material information of river segments
INPUT	pihm-geol	Geologic file
INPUT	pihm-ibc	Boundary condition information for elements
INPUT	pihm-modelinfo	PIHM model information aggreagation file
INPUT	pihm-lc	Vegetation parameters of different land cover types
INPUT	pihm-base	Base file
INPUT	pihm-forc	PIHM forcing file with the majority of the relevant variables
INPUT	pihm-soil	Soil parameters for the soil types
INPUT	pihm-att	PIHM attribute file with index values of variables for timeseries
OUTPUT	pihm-et0	Evaporation canopy file
OUTPUT	pihm-rivFlx9	lateral outflux to the bed beneath river
OUTPUT	pihm-rivFlx4	Baseflow to stream reach from aquifer on the left
OUTPUT	pihm-rech	Recharge Rate file
OUTPUT	pihm-rivFlx10	lateral influx to the bed beneath river
OUTPUT	pihm-infiltration	Infiltration file

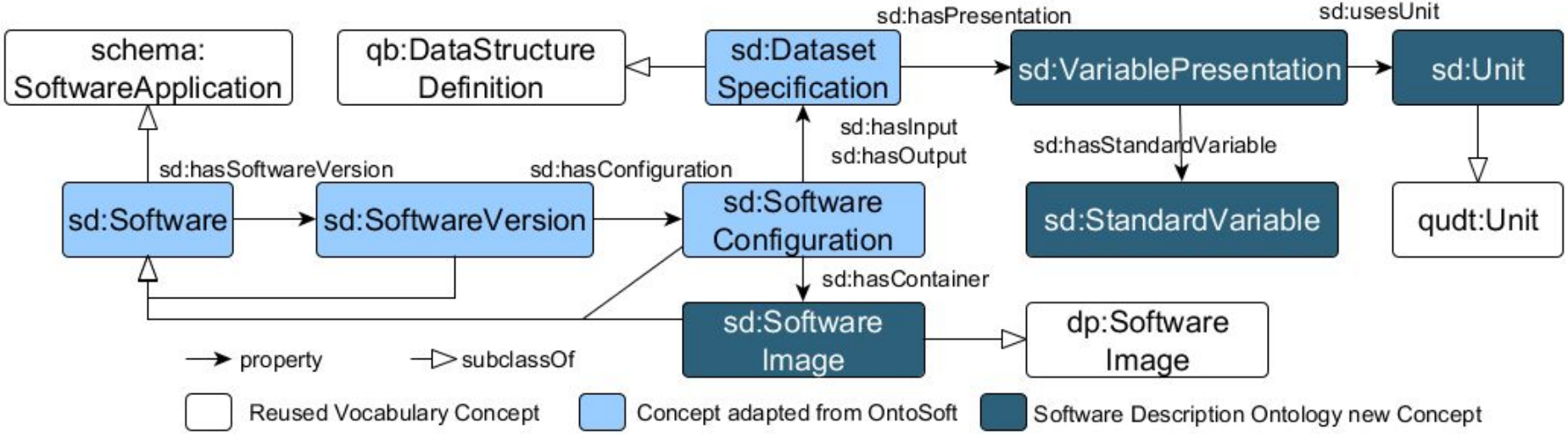
pihm-riv

Input/output variables

ormation of river segments

Label	Long Name	Description	Standard Name	Units
Bed	Bed Depth	Bed Depth	channel_bed__thickness	m
KsatV	Bed Hydraulic Conductivity	Bed Hydraulic Conductivity	soil_water__vertical_saturated_hydraulic_conductivity	m day-1
Water table value	Water table of the IC	Water table of the IC		m

Software Description Ontology



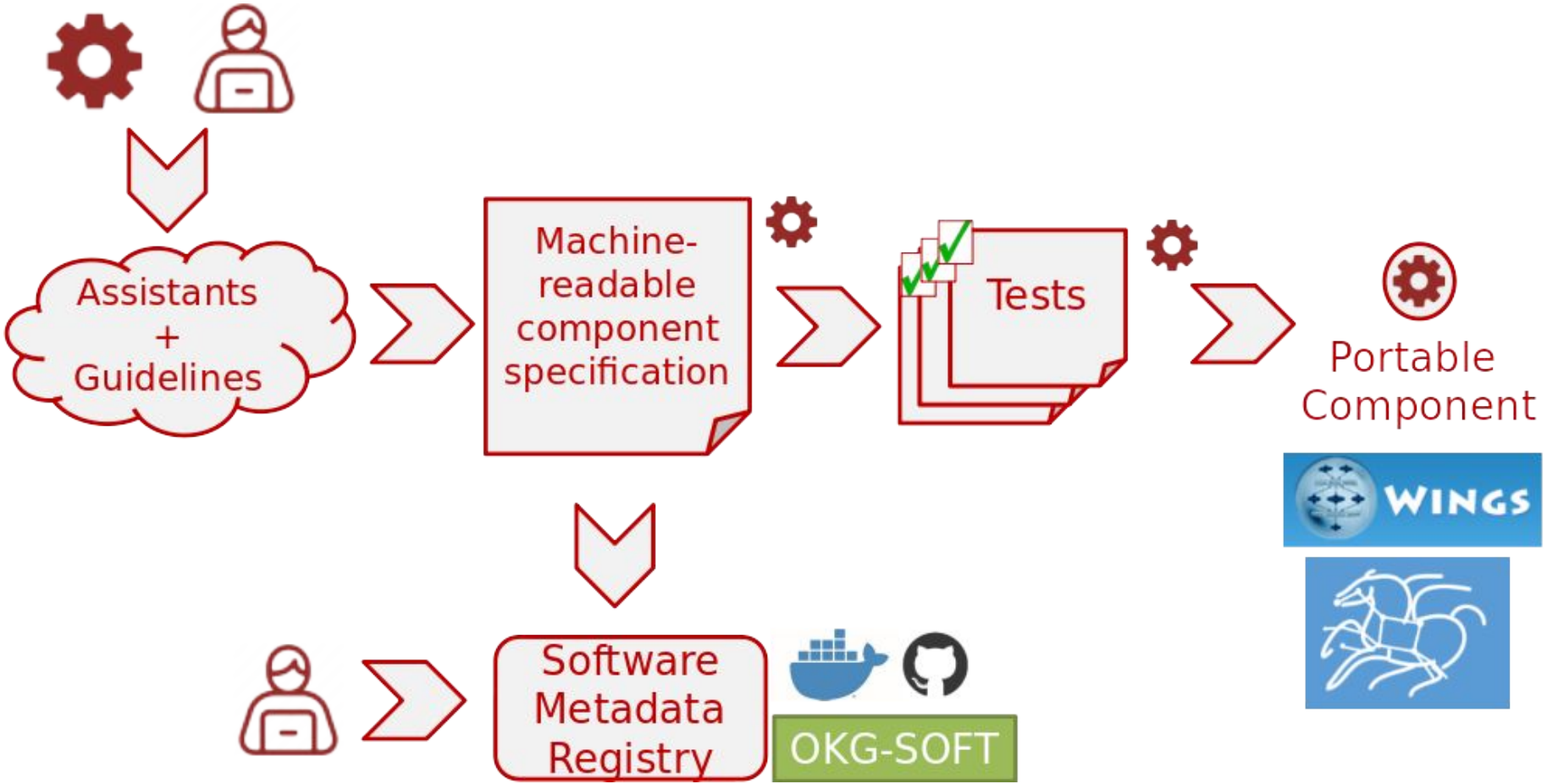
Extending:

- Schema.org/Codemeta (software metadata)
- W3C Data Cubes (Contents of inputs and outputs)
- NASA QUDT (Units)
- DockerPedia (Software images)
- Scientific Variables Ontology (Standard Variables)

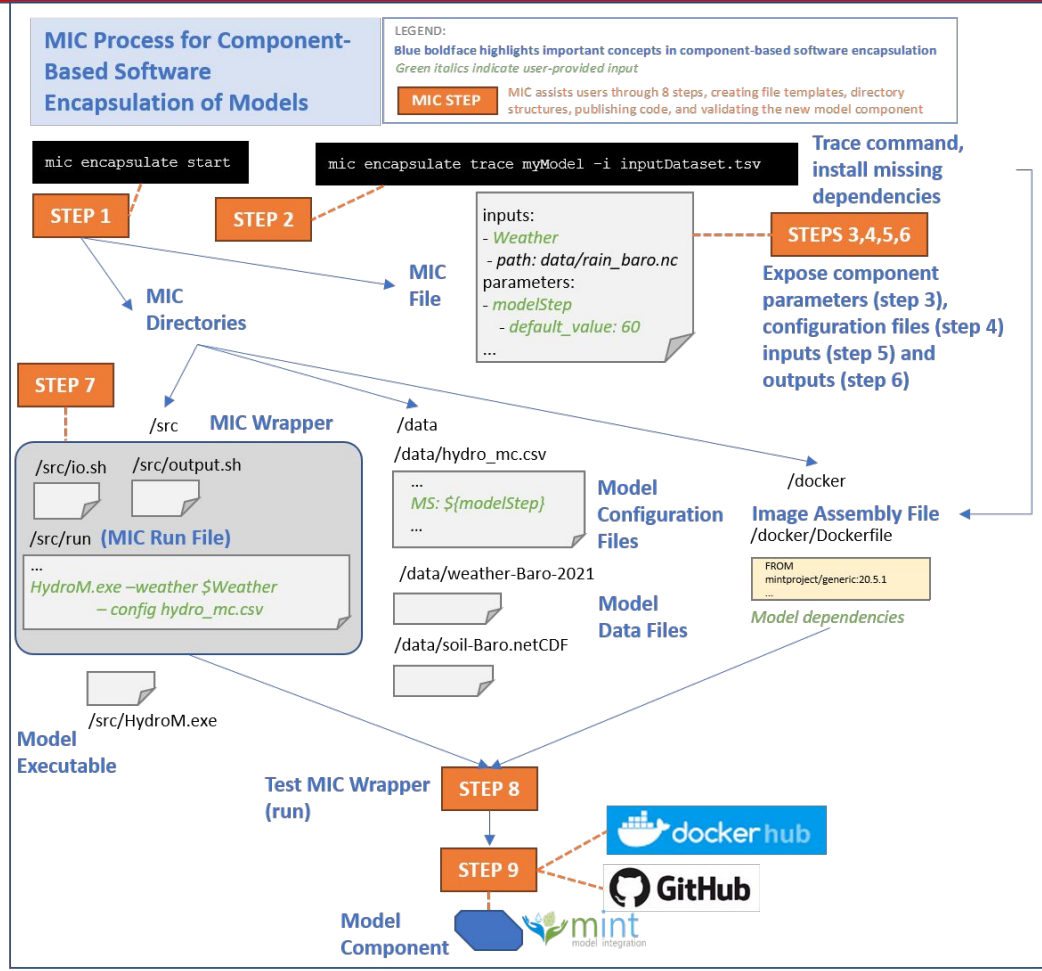
https://w3id.org/okn/o/sd

Garijo, D., Osorio, M., Khider, D., Ratnakar, V., & Gil, Y. (2019, September). OKG-Soft: An open knowledge graph with machine readable scientific software metadata. In *2019 15th International Conference on eScience (eScience)* (pp. 349-358). IEEE.

Scientific Software encapsulation



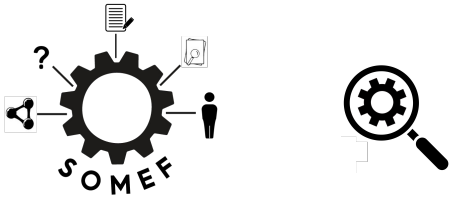
Reusing software: model encapsulation



Software encapsulation methodology

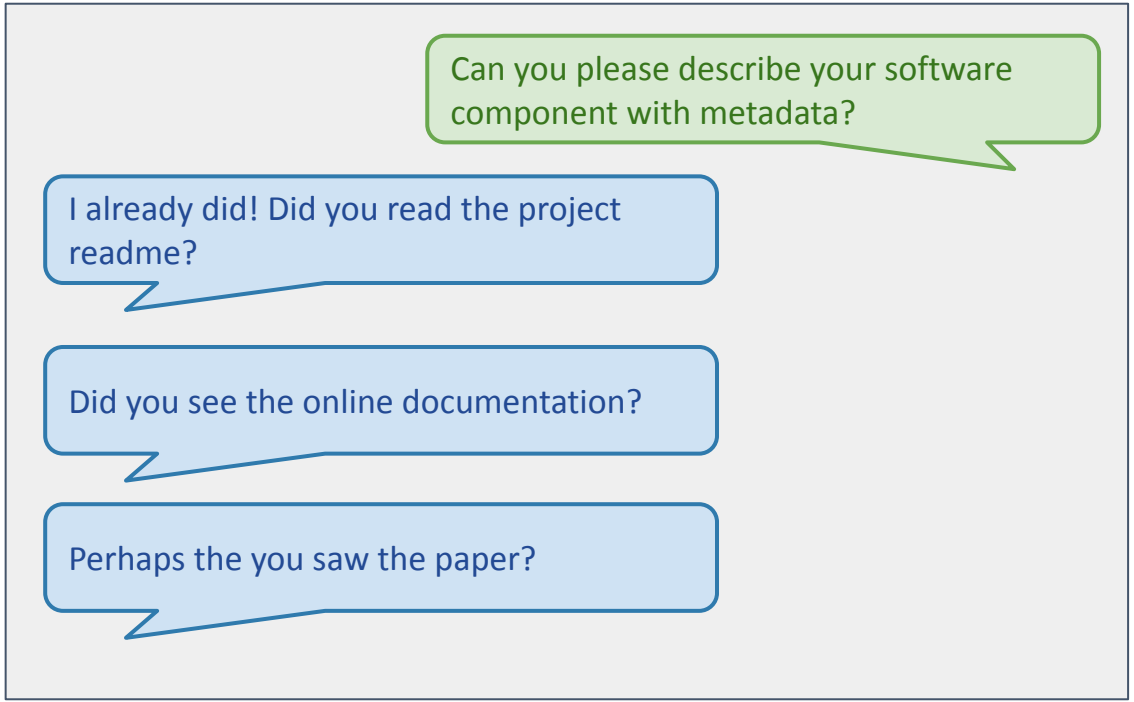
- Input: software component
- Output:
  - Docker image
  - Wrapper script (GitHub)
  - Metadata (MINT model catalog)
- Powered by ReproZip (https://www.reprozip.org/) to automatically suggest I/O

SOMEF: Automating software metadata extraction



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

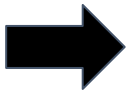
Software metadata is not ~~abundant~~ machine readable

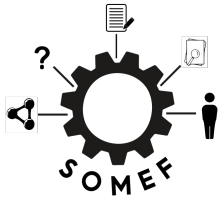


Many domain-specific registries are curated by hand by experts

SOMEF: Automating Software Metadata Extraction

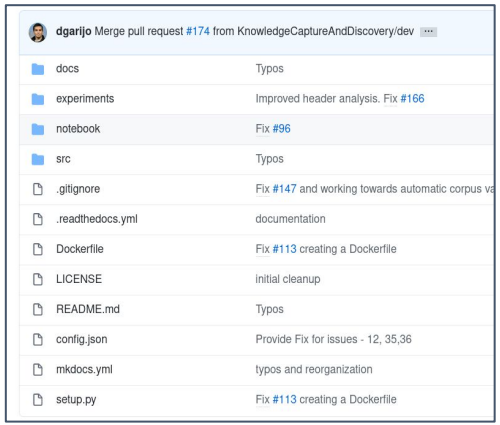
Repository





Extraction

Results (Metadata)



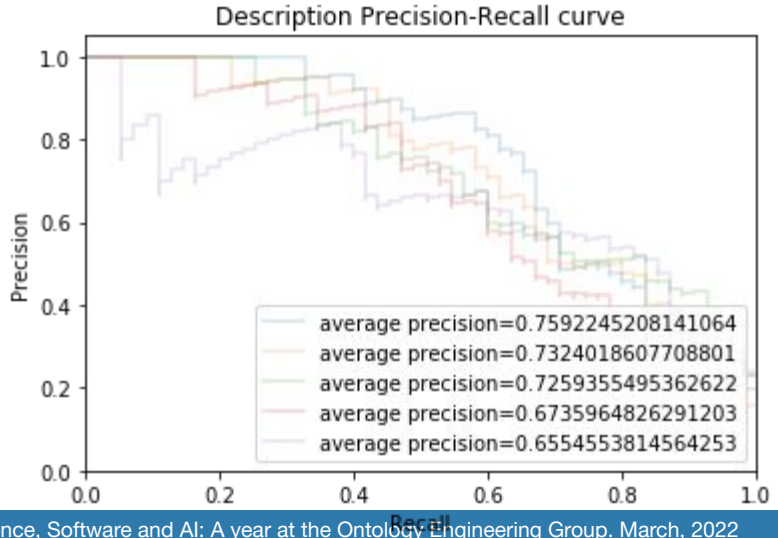
- **Readme Analysis**
  - Supervised classification
  - Regular expressions
  - Header analysis
- **File exploration**
  - Notebooks
  - Dockerfiles
  - Documentation
- **GitHub API**

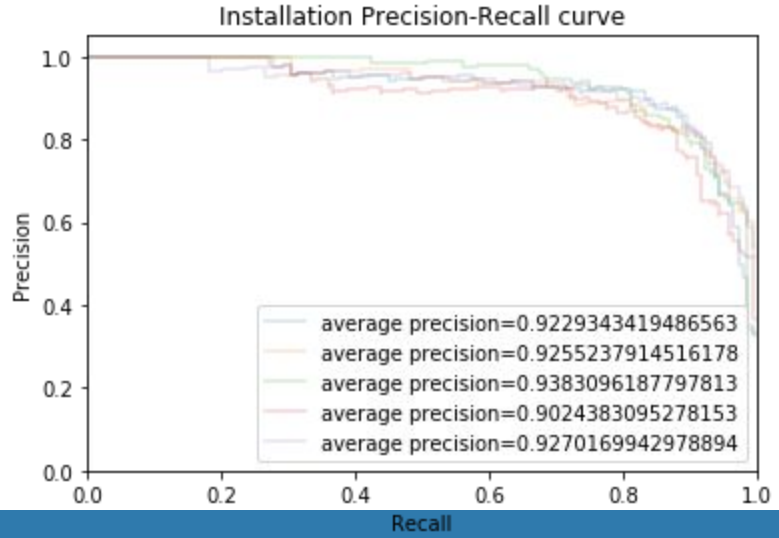
CodeMeta

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

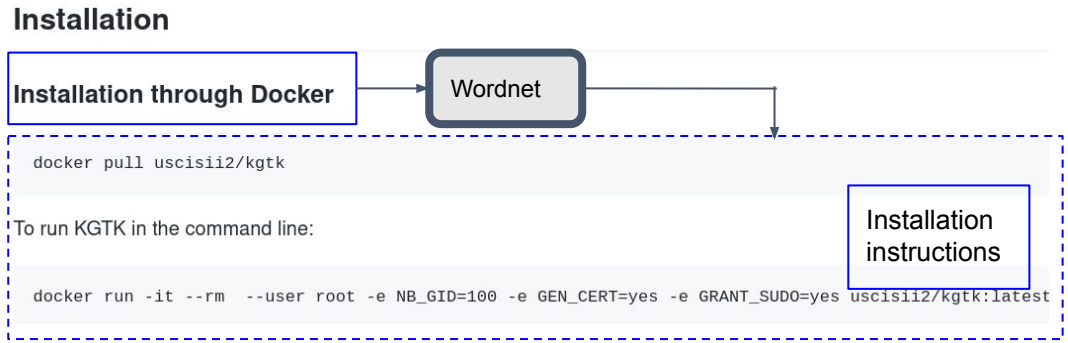
- Paragraph-based text classification
- Four main categories:
  - Installation, citation, description, invocation.
- Binary classification problem

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





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



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

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:

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)
- **Stars** (GA)
- **Contact** (HA)
- **Download URL** (HA, GA)
- **DOI** (RE)
- **DockerFile** (FE)
- **Notebooks** (FE)
- **Executable notebooks (Binder)** (RE)
- **Owner**: (GA)
- **Keywords** (GA)
- **Source code** (GA)
- **Releases** (GA)
- **Changelog** (GA)
- **Issue tracker** (GA)
- **Programming languages** (GA)
- **Acknowledgements** (HA)

Method used:

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

Example Output

CodeMeta


```
description:
  0:
    excerpt:      "WIDOCO helps you to publish and create an enriched and customized documentation of you
                  classes, properties and data properties of the ontology, the OOPS! webservice by María
                  being used. In addition, we use WebVowl to visualize the ontology and have extended Bub
                  documentation of the terms in your ontology (based on [LODE](http://www.essepuntato.it/
                  annotation in JSON-LD snippets of the html produced.\n* Association of a provenance pag
                  means to complete it on the fly when generating your ontology. Check the [best practice
                  WIDOCO.\n* Guidelines on the main sections that your document should have and how to co
                  changelog of differences between the actual and the previous version of the ontology (b
                  them independently and replace only those needed.\n* Content negotiation and serializat
    confidence:
      0:          1
    technique:    "wordnet"
  1:
    excerpt:      "For a complete list of the current improvements and next features, check the project o
    confidence:
      0:          0.8231493588525339
    technique:    "classifier"
  2:
    excerpt:      "Wizard for documenting ontologies. WIDOCO is a step by step generator of HTML template
    confidence:
      0:          1
    technique:    "metadata"
citation:
  0:
    excerpt:      "@inproceedings{garijo2017widoco,\n  title={WIDOCO: a wizard for documenting ontologies
                  organization={Springer, Cham},\n  doi = {10.1007/978-3-319-68204-4_9},\n  funding = {US
    confidence:
      0:          1
    technique:    "classifier"
```

```
@context:       "https://doi.org/10.5063/schema/codemeta-2.0"
@type:          "SoftwareSourceCode"
license:        "https://raw.githubusercontent.com/dgarijo/Widoco/master/LICENSE"
codeRepository: "git+https://github.com/dgarijo/Widoco.git"
dateCreated:    "2013-07-15"
datePublished:  "2020-12-14"
dateModified:   "2021-03-16"
downloadUrl:    "https://github.com/dgarijo/Widoco/releases"
issueTracker:   "https://github.com/dgarijo/Widoco/issues"
name:           "Widoco"
version:        "v1.4.15_1"
description:
  0:            "Wizard for documenting ontologies. WIDOCO is a step by step generat
  1:            "WIDOCO helps you to publish and create an enriched and customized d
                the classes, properties and data properties of the ontology, the OOP
                URI and title being used. In addition, we use WebVowl to visualize t
                WIDOCO:\n* Automatic documentation of the terms in your ontology (ba
                /index.html))\n* Automatic annotation in JSON-LD snippets of the htm
                extraction from the ontology plus the means to complete it on the fl
                to know more about the terms recognized by WIDOCO.\n* Guidelines on
                (http://vowl.visualdataweb.org/webvowl/)).\n* Automatic changelog of
                /)).\n* Separation of the sections of your html page so you can writ
                practices\n\n"
  2:            "For a complete list of the current improvements and next features,
releaseNotes:   "This pre-release fixes issues regarding namespace prefixes (now the
                settings in your visualization)\r\n\r\nMore information on the addre
keywords:
  0:            "ontology"
  1:            "wizard"
  2:            "metadata"
  3:            "documentation"
  4:            "ontology-diagram"
  5:            "ontology-evaluation"
```

A year at the Ontology Engineering Group…

Research Projects

Static code analysis in Python

- Extraction of available classes and functions
  - Documentation
- Requirements (reusing existing libraries)
- Call list
- File hierarchy
- Control flow (reusing existing libraries)
- Software invocation
  - Service? Package? Library? & invocation command
- Metadata export in JSON

Inspect4py

Rosa Filgueira

University of St Andrews

Benefits

- Understanding, reuse, ML featurization, similarity, best practices

Filgueira, R. and Garijo, D. (2022). Inspect4py: A Knowledge Extraction Framework for Python Code Repositories. To appear in Mining Software Repositories, 2022 (demo)

Can we extract additional context from READMEs?

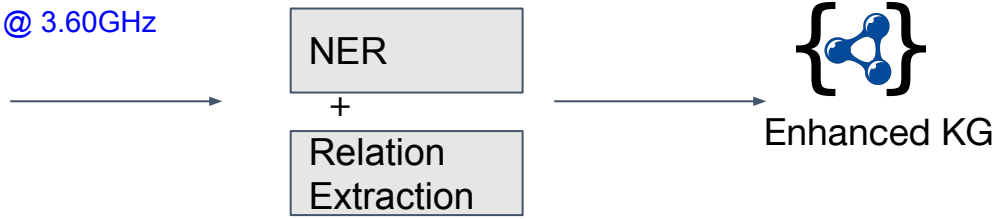
- E.g., installation instructions for Unix/Windows
- Hardware requirements and versions
- Library and OS versions
- Funding

Ontology Engineering Group

Pablo Calleja
Marco Santucci (NER)
Andres Cardozo (RE)

Requirements (Unix)
CPU: Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz
GPU: NVIDIA GTX1060
python 2.7:

- pytorch == 0.3.1
- numpy
- opencv



https://github.com/oeg-upm/ner4soft/

Awesome lists capture common software projects under a common theme

- Can we use READMEs as labels for weak supervised classification?
- Text-based classification + some deep learning architectures (Bi-lstm)

Jenifer Ciuiu-Kiss

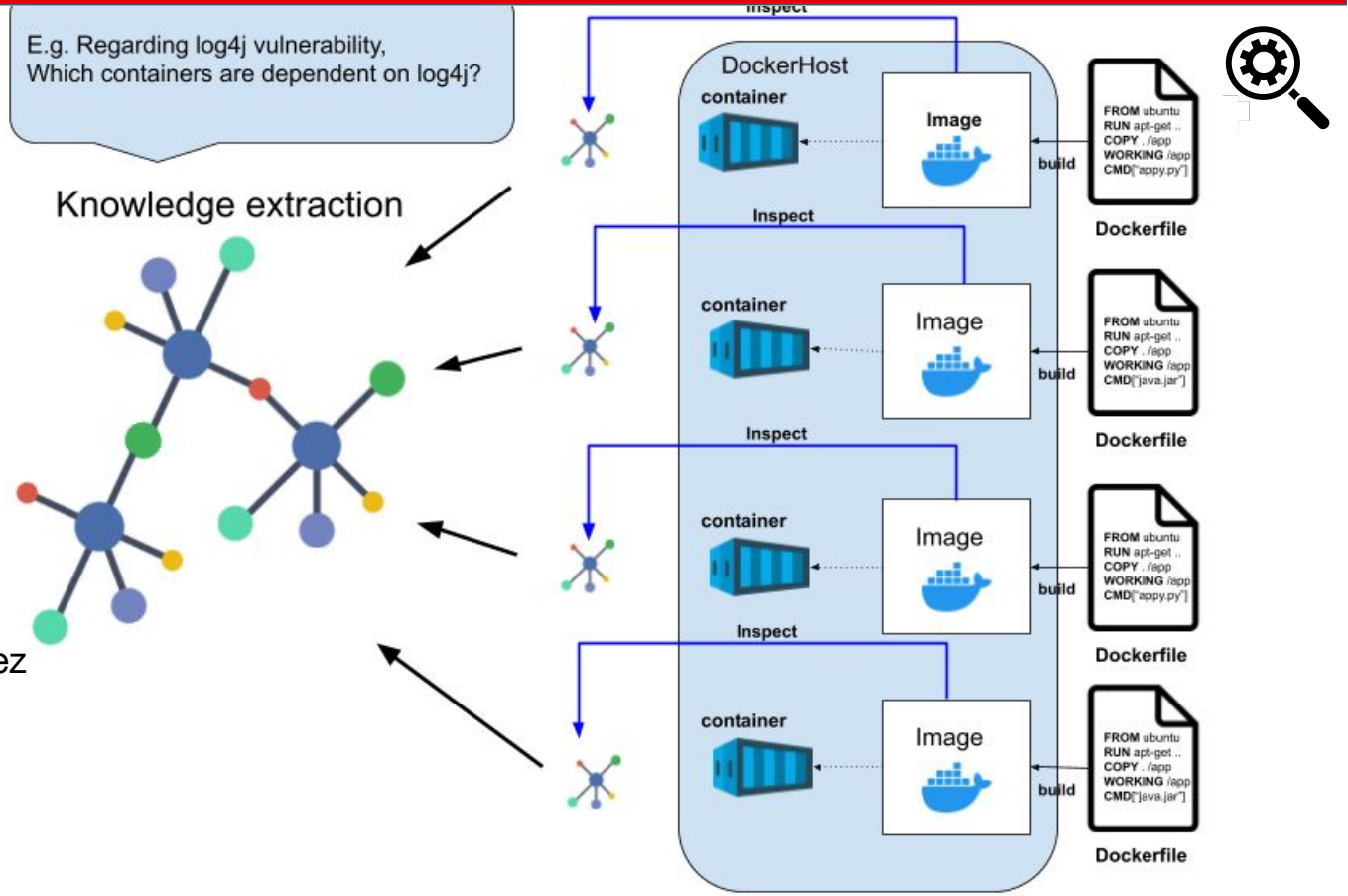
WIDOCO → “Semantic Web”

PYLEOCLIM → “Geosciences”

Extracting knowledge from other sources related to software (TFG)

Can we extract useful metadata from Docker images and Jupyter notebooks to gain additional insight in a repository?

Jhon Toledo (Docker)
Daniel Rodriguez (notebooks)



Osorio, M., Buil-Aranda, C., Santana-Perez, I., & Garijo, D. (2022). DockerPedia: A Knowledge Graph of Software Images and Their Metadata. *International Journal of Software Engineering and Knowledge Engineering*, *32*(01), 71-89.

How can we identify similarities between software components?

- Can we identify differences?
- What does similarity mean?
  - Metadata?
  - Description?
  - Documentation?
  - Functions?
  - KG diff?
  - Code?

Clark Wang

Topic modeling (TFM)

María Ayuso

- Are readme files appropriate for topic modeling-based similarity?
  - LDA
  - BERT-topics
  - Comparison against human-tagged categories

A year at the Ontology Engineering Group…

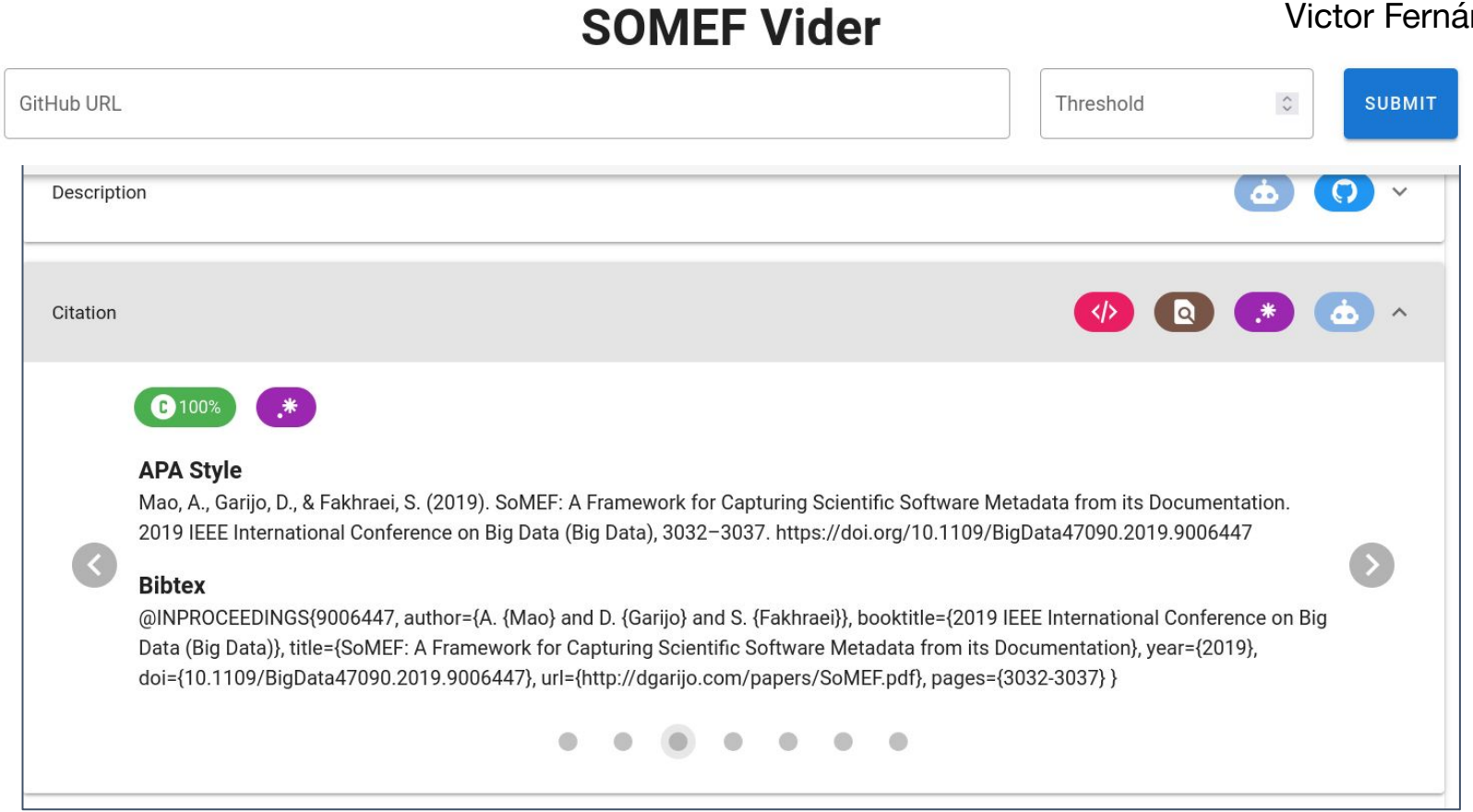
Engineering and demo projects

New (engineering) features in SOMEF

- Better parsing
- Pypi support
- Gitlab support
- New metadata fields
  - Colab notebooks
  - Status
  - sh scripts
  - logos
  - images
  - code of conduct
  - arxiv links
  - …
- Additional types of files
- Better provenance (which technique was used and how)
- Robustness

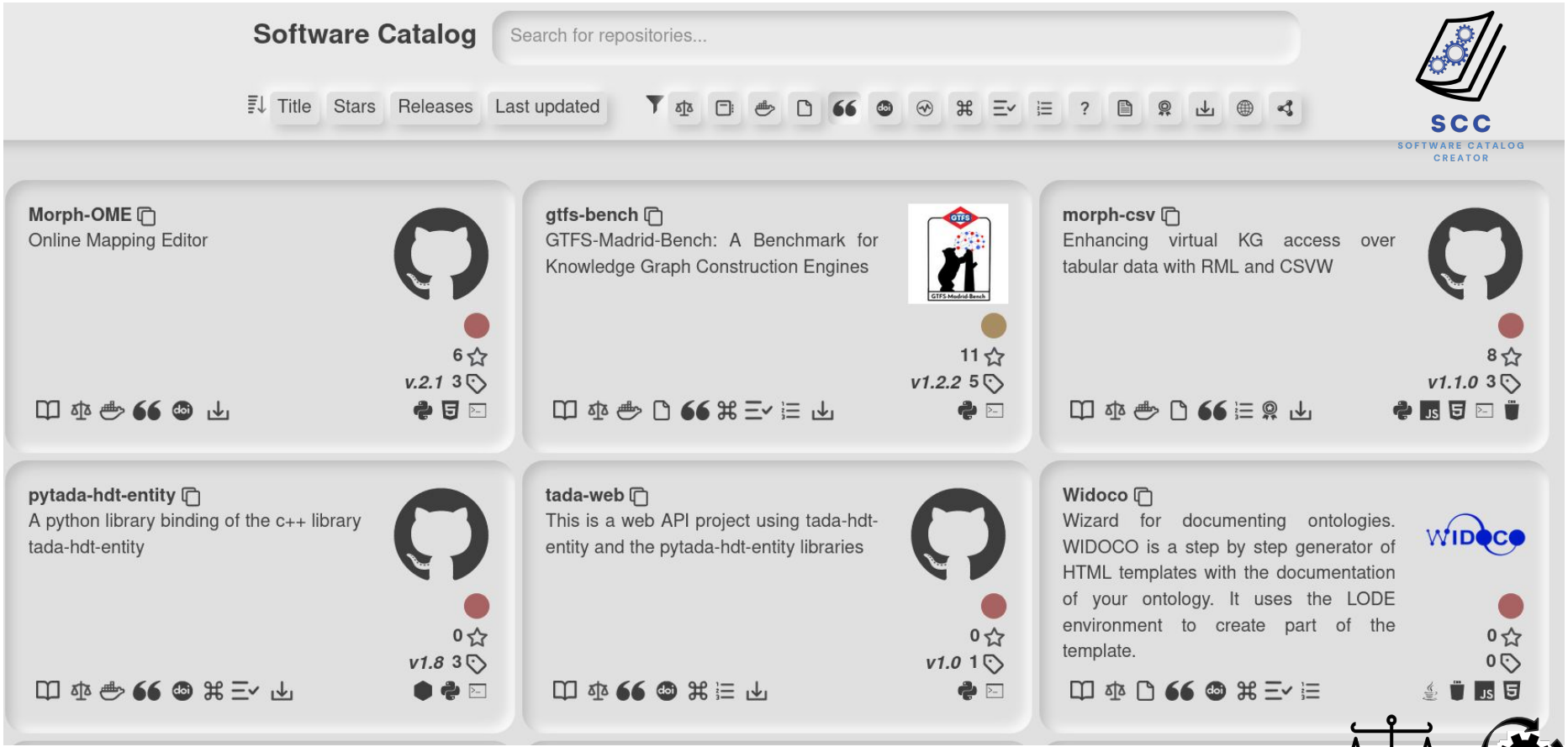
Miguel Ángel García

Victor Fernández



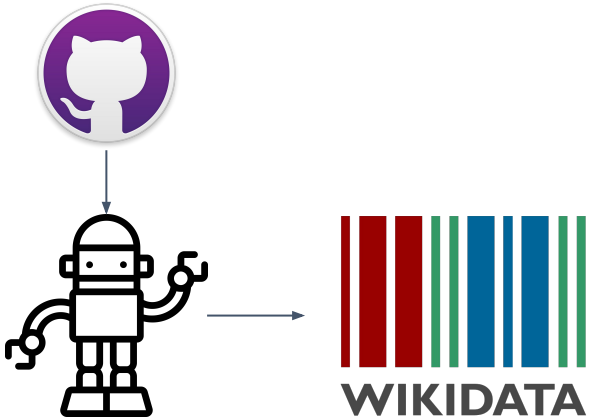
https://somef.linkeddata.es/

New OEG Software Catalog (TFG)

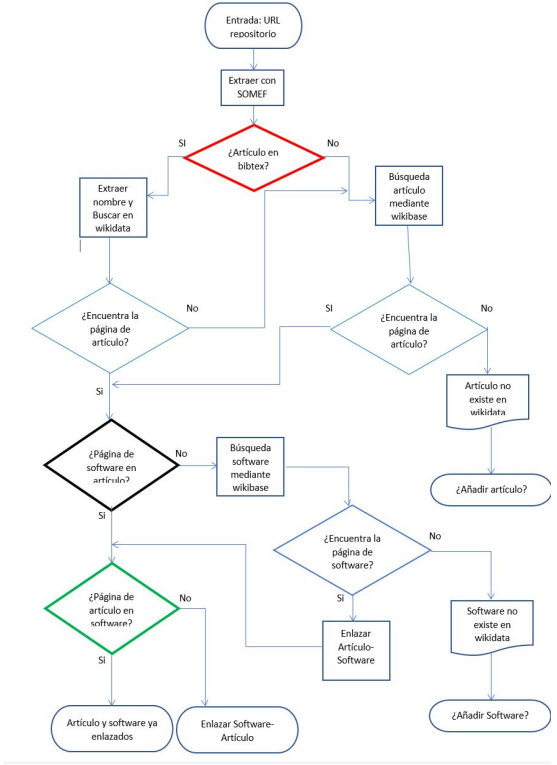


Daniel Rodriguez alpha available at: https://dakixr.github.io/scc/example/index.html

Wikidata **bot** to link code repositories with Wikidata articles



Article exists in WD?
Software exists in WD?
Are they linked?



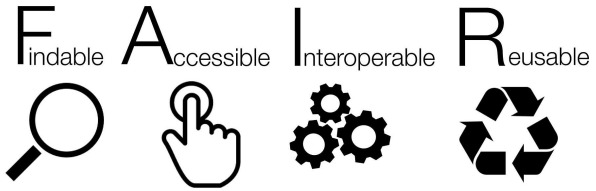
Jorge Bolinches

Summing up

Research software is a critical asset for Open Science

Accelerating Software Understanding requires:

- Automated description
- Assisted comparison
- Easy reuse



Additional challenges:

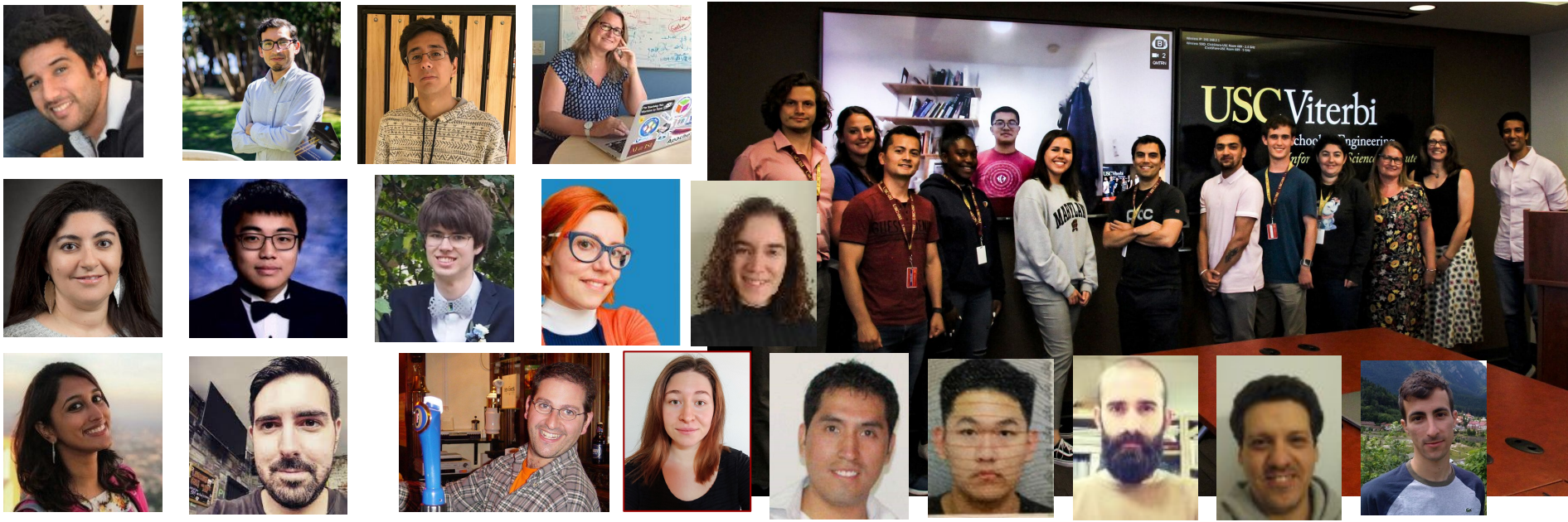
- Relation extraction between research outputs

- Create comparison benchmarks


- Representation of software metadata in RDF-star

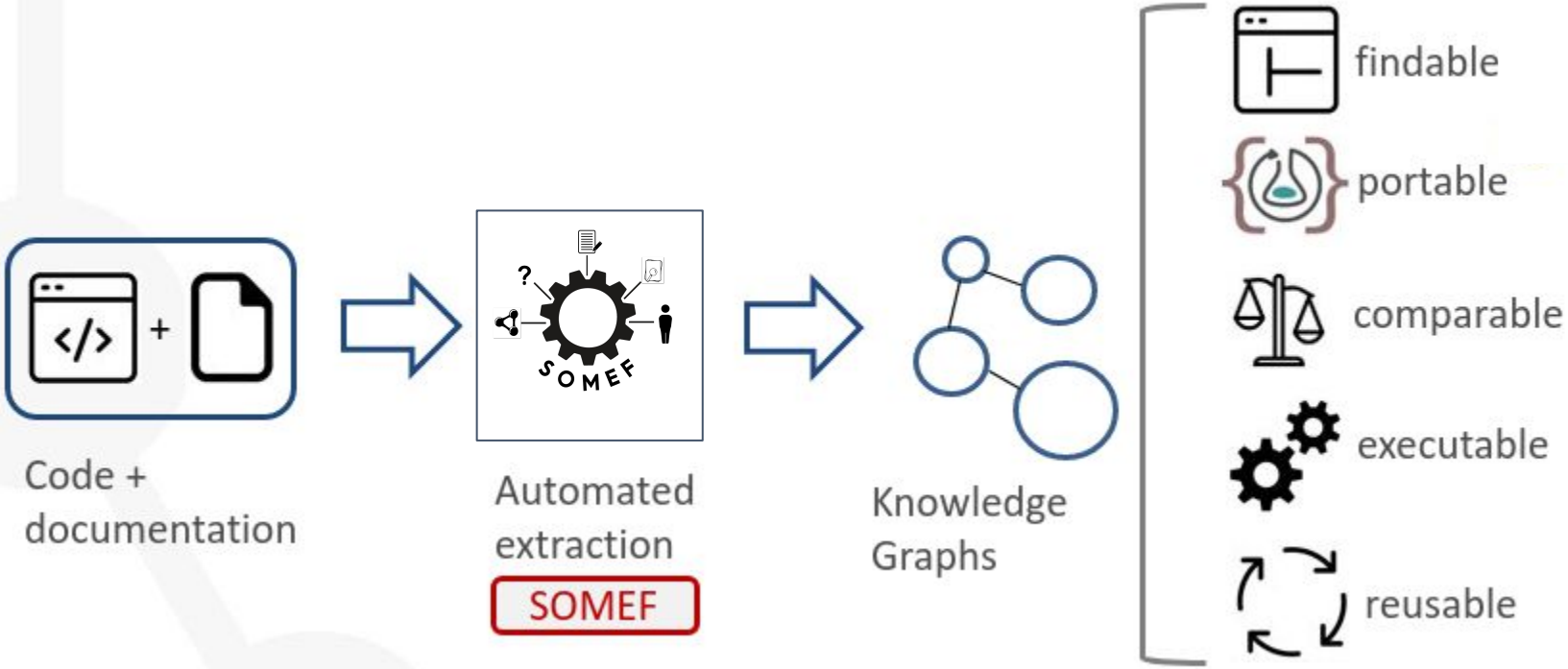
- Can an assistant follow your installation instructions?
  - Tests?

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 & 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)



Let's create **machine-actionable** software metadata