Improving FAIRness with containers

Ana Trisovic
IQSS, Harvard University

SORSE

A Series of Online Research Software Events, Oct 7, 2020

Agenda

- A quick summary of FAIR principles
- The new role of data repositories
- FAIR in practice - a code rerunability study
- Overview of new tools
- A new solution

A quick summary of FAIR principles

Findable	
Accessible	
Interoperable	
Reusable	

A quick summary of FAIR principles

Findable	Describe data in metadata, assign DOI Metadata record is shared in data repository
Accessible	Accessible but not necessarily open Standard access protocol
Interoperable	File format open or proprietary Description of data elements
Reusable	License and usage rights Data provenance

Wilkinson, Mark D., et al. "The FAIR Guiding Principles for scientific data management and stewardship." *Scientific data* (2016)

How data repositories incorporate FAIR principles

- Dataverse - open-source research data repository software
  - Mandatory citation-level metadata, with DOI
  - Rich metadata (including domain-specific)
  - Six levels of data access (open and sensitive)
  - Compliance with community standards
  - Data exploration and external tools, etc.

New role of data repositories

- Research code is often deposited with data
- Typically to enable verification and reproducibility of results from published papers
- There are 2200+ datasets that contain Python or R code only at Harvard Dataverse.

FAIR principles and software best practices

Findable	Describe code in metadata, provide versions, identifiers, contributors, citations etc.
Accessible	Make source code open and publicly accessible from day one
Interoperable	Share code metadata in a community registry
Reusable	Adopt a license

Jiménez, Rafael C., et al. "Four simple recommendations to encourage best practices in research software." *F1000Research* (2017)

Applying FAIR principles for code

Findable	Describe code in metadata, assign DOI for all versions, add it searchable software registry
Accessible	Access protocol free, open, universal, allows authentication, metadata available
Interoperable	Use of broadly applicable language to facilitate machine readability, document dependencies
Reusable	Usage licenses, add provenance, code metadata and documentation to meet community standards

Lamprecht, Anna-Lena, et al. "Towards FAIR principles for research software." *Data Science* Preprint (2019)

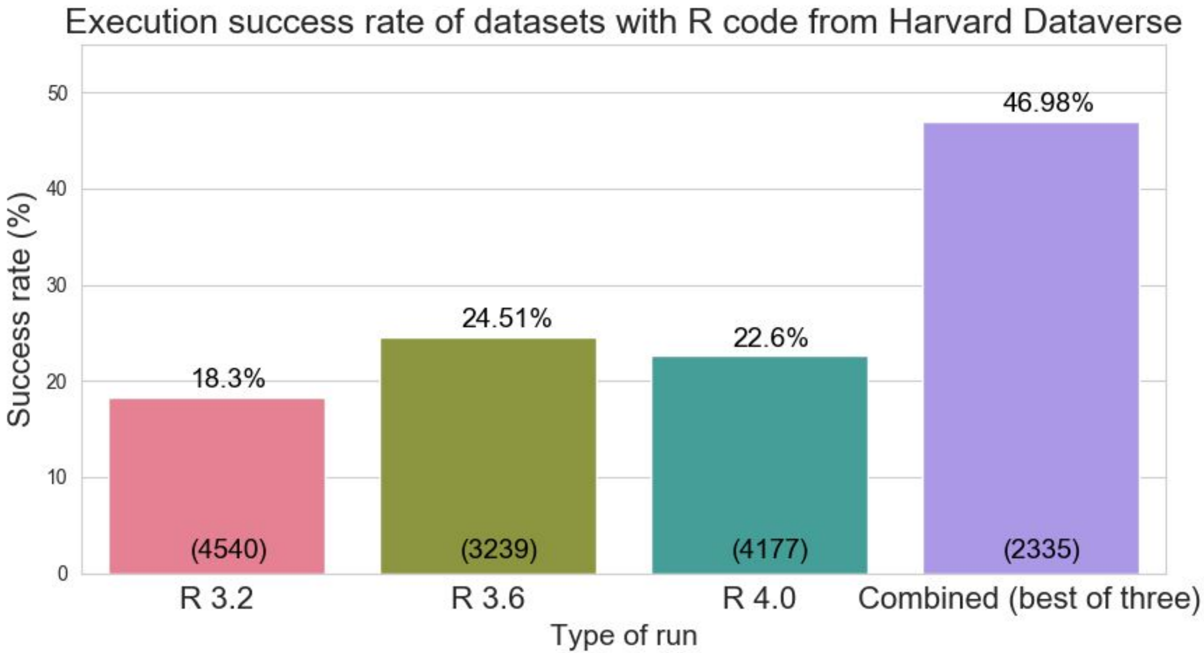
Feasible FAIRness for research code

- Code metadata
- Licenses for code reuse
- Document code dependencies

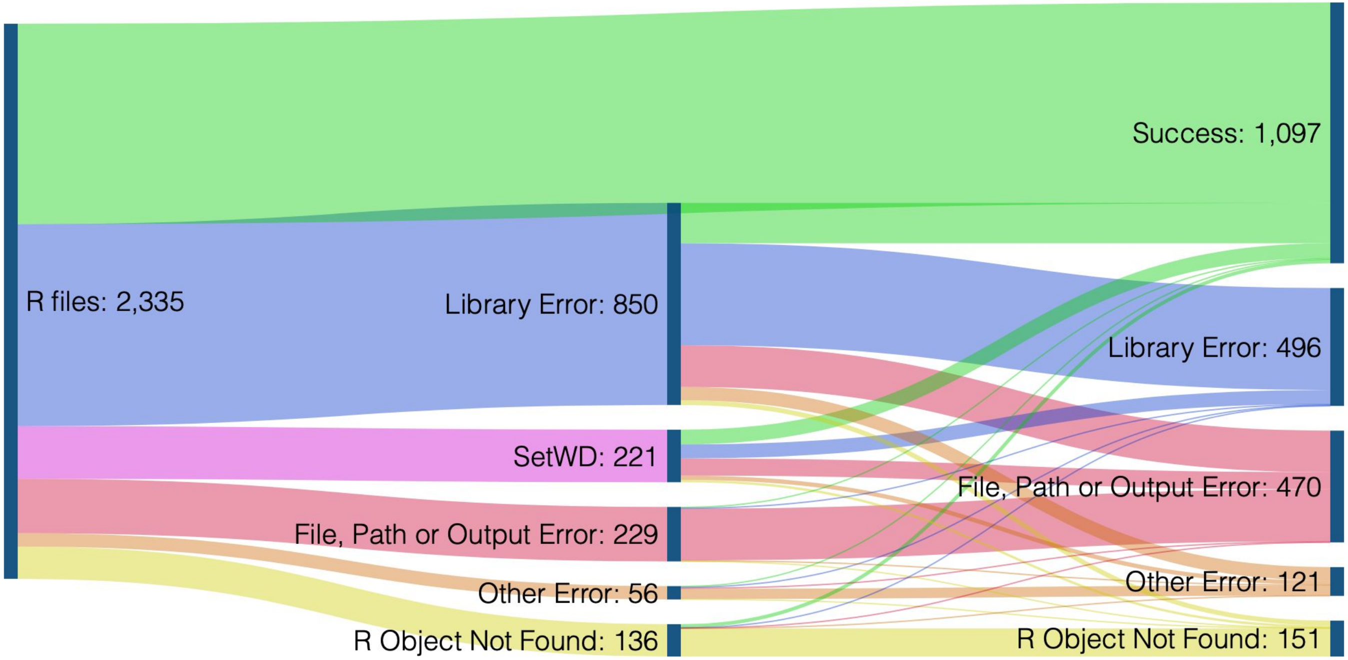
What’s happening in practice?

- What happens when a researcher downloads data and code, pre-installs all code dependencies and tries to rerun it
- We simulate this workflow on AWS, where one Dataverse dataset is allocated up to 5 hours to run and then, we record a result
- Note: Not a reproducibility study!

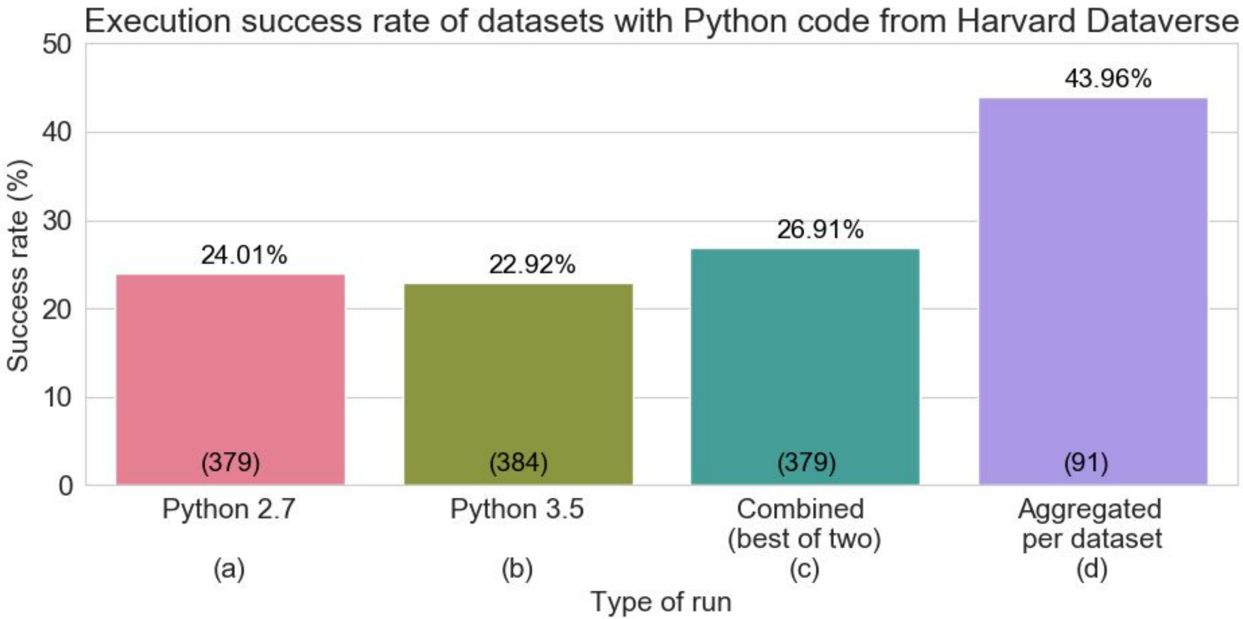
Results with R code from Dataverse



Most common errors



Python results



What do these results tell us?

- Code is not easily reusable
  - R and Python are not always backward compatible
  - Rerunnability when requirements is present
  - Fixed paths are common
- Lack of support for support for code dependencies

Virtual machines and containers

- Capture necessary system dependencies and can vastly improve reproducibility and code rerunability
- Portable and shareable
- New tools based on virtual containers

ReproZip

Stencila

CODE OCEAN

WHOLE TALE

RENKU

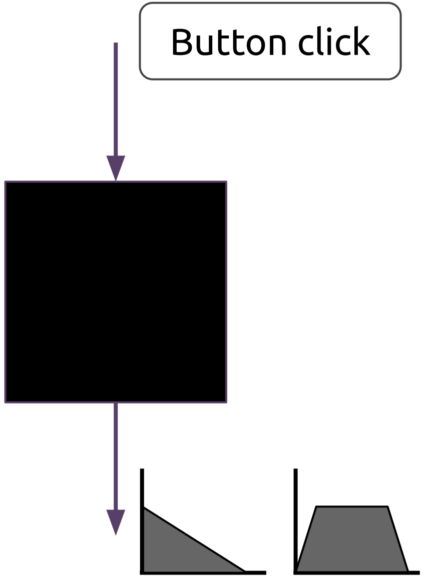
A FAIR black box in data repositories

- A FAIR solution: Store exported container image files in data repository
- With good metadata that documents all that is inside - it is FAIR

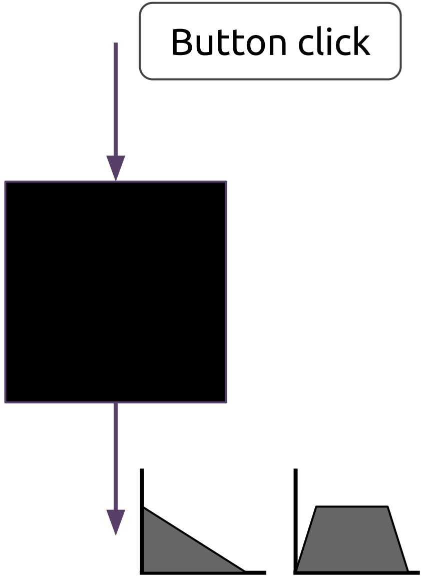
A FAIR black box in data repositories

- A FAIR solution: Store exported container image files in data repository
- With good metadata that documents all that is inside - it is FAIR

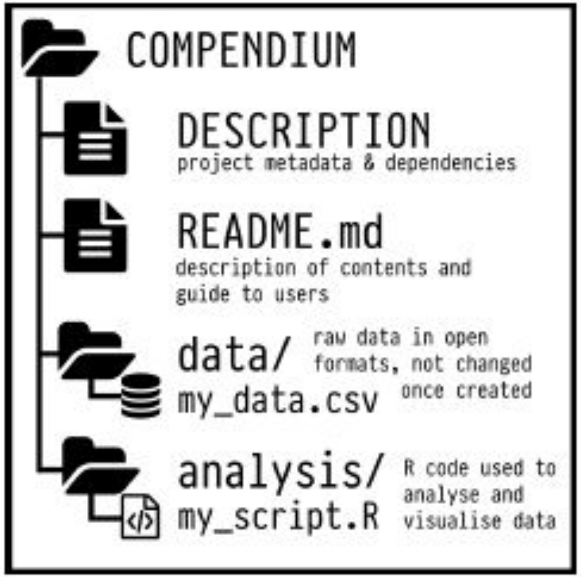
Reproducible versus reusable



Reproducible versus reusable



Button click



COMPENDIUM

DESCRIPTION
project metadata & dependencies

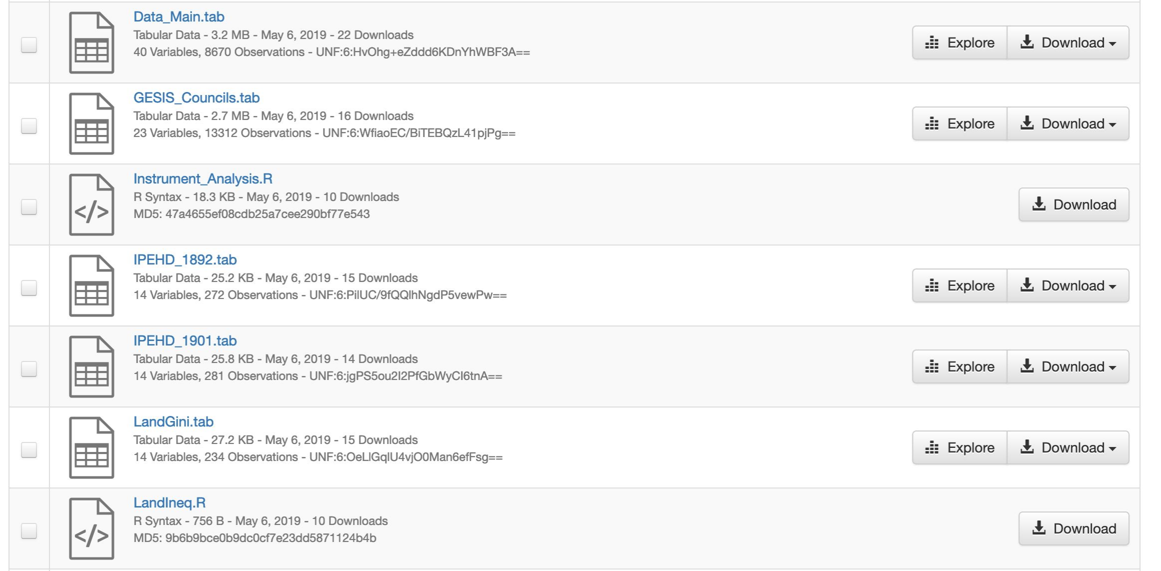
README.md
description of contents and guide to users

data/
my_data.csv
raw data in open formats, not changed once created

analysis/
my_script.R
R code used to analyse and visualise data

Transparency and reusability

- Value in viewing research data and code from a browser



□ Data_Main.tab
Tabular Data - 3.2 MB - May 6, 2019 - 22 Downloads
40 Variables, 8670 Observations - UNF:6:HvOhg+eZddd6KDnYhWBF3A==

□ GESIS_Councils.tab
Tabular Data - 2.7 MB - May 6, 2019 - 16 Downloads
23 Variables, 13312 Observations - UNF:6:WfiaoEC/BiTEBQzL41pjPg==

□ Instrument_Analysis.R
R Syntax - 18.3 KB - May 6, 2019 - 10 Downloads
MD5: 47a4655ef08cdb25a7cee290bf77e543

□ IPEHD_1892.tab
Tabular Data - 25.2 KB - May 6, 2019 - 15 Downloads
14 Variables, 272 Observations - UNF:6:PilUC/9fQQlhNgdP5vewPw==

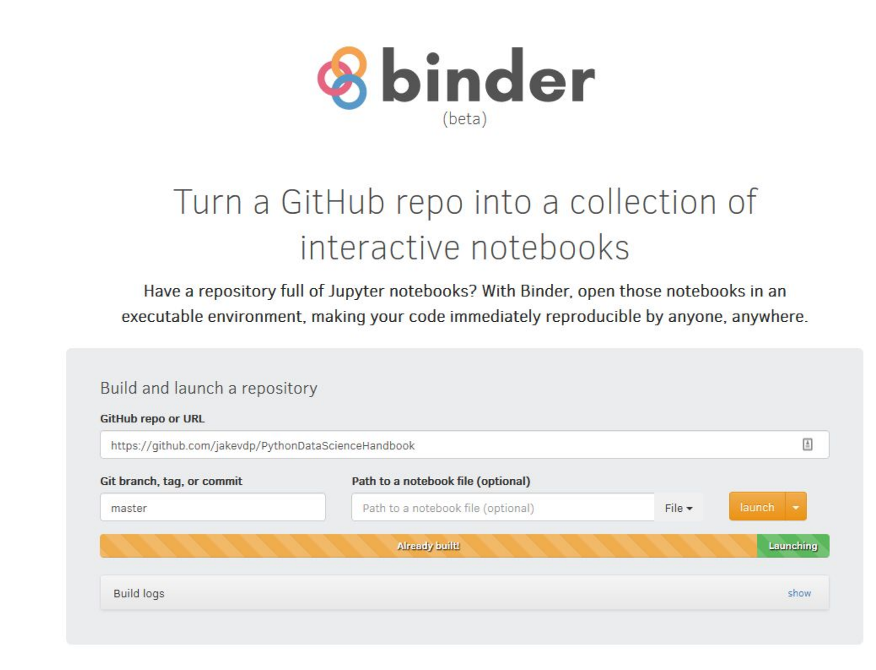
□ IPEHD_1901.tab
Tabular Data - 25.8 KB - May 6, 2019 - 14 Downloads
14 Variables, 281 Observations - UNF:6:jgPS5ou2I2PfGbWyCI6tnA==

□ LandGini.tab
Tabular Data - 27.2 KB - May 6, 2019 - 15 Downloads
14 Variables, 234 Observations - UNF:6:OeLlGqlU4vjO0Man6efFsg==

□ LandIneq.R
R Syntax - 756 B - May 6, 2019 - 10 Downloads
MD5: 9b6b9bce0b9dc0cf7e23dd5871124b4b

Improving FAIRness with cutting-edge tools

- Jupyter Binder
- Automatically-generated elaborate Dockerfiles (100+ lines) that will stand a test of time



Turn a GitHub repo into a collection of interactive notebooks

Have a repository full of Jupyter notebooks? With Binder, open those notebooks in an executable environment, making your code immediately reproducible by anyone, anywhere.

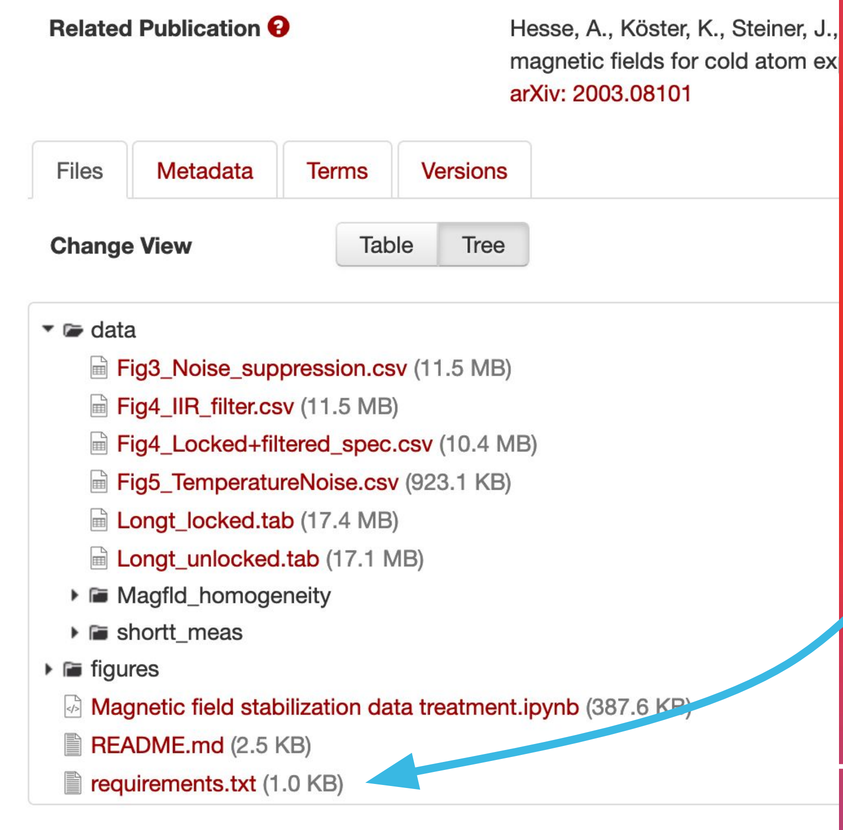
Build and launch a repository

GitHub repo or URL: https://github.com/jakevdp/PythonDataScienceHandbook

Git branch, tag, or commit: master

Path to a notebook file (optional): Path to a notebook file (optional)

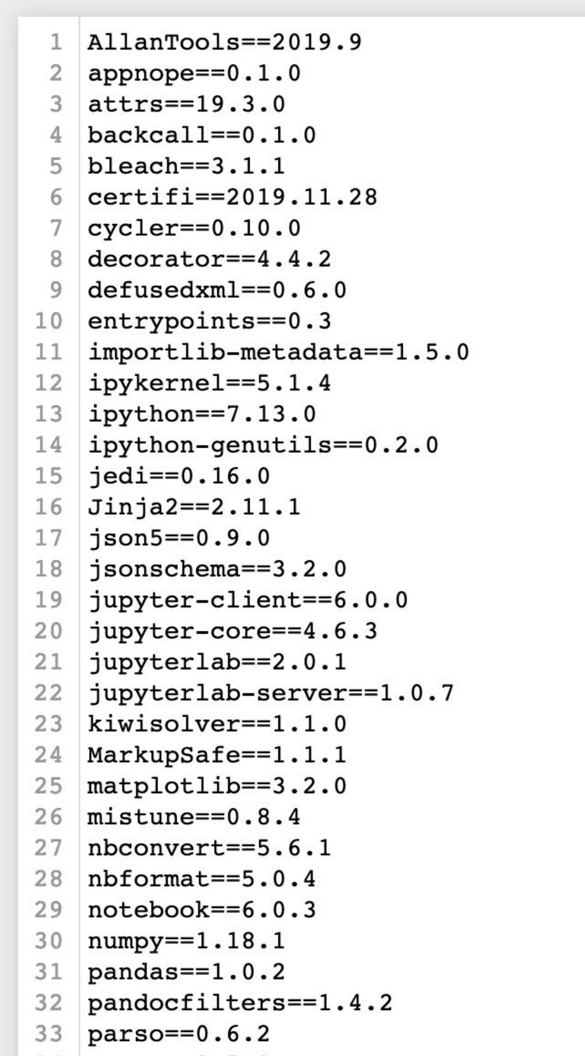
“ How does this work?



Related Publication ❓

Hesse, A., Köster, K., Steiner, J.,
magnetic fields for cold atom ex
arXiv: 2003.08101

- data
  - Fig3_Noise_suppression.csv (11.5 MB)
  - Fig4_IIR_filter.csv (11.5 MB)
  - Fig4_Locked+filtered_spec.csv (10.4 MB)
  - Fig5_TemperatureNoise.csv (923.1 KB)
  - Longt_locked.tab (17.4 MB)
  - Longt_unlocked.tab (17.1 MB)
  - Magfld_homogeneity
  - shortt_meas
- figures
- Magnetic field stabilization data treatment.ipynb (387.6 KB)
- README.md (2.5 KB)
- requirements.txt (1.0 KB)

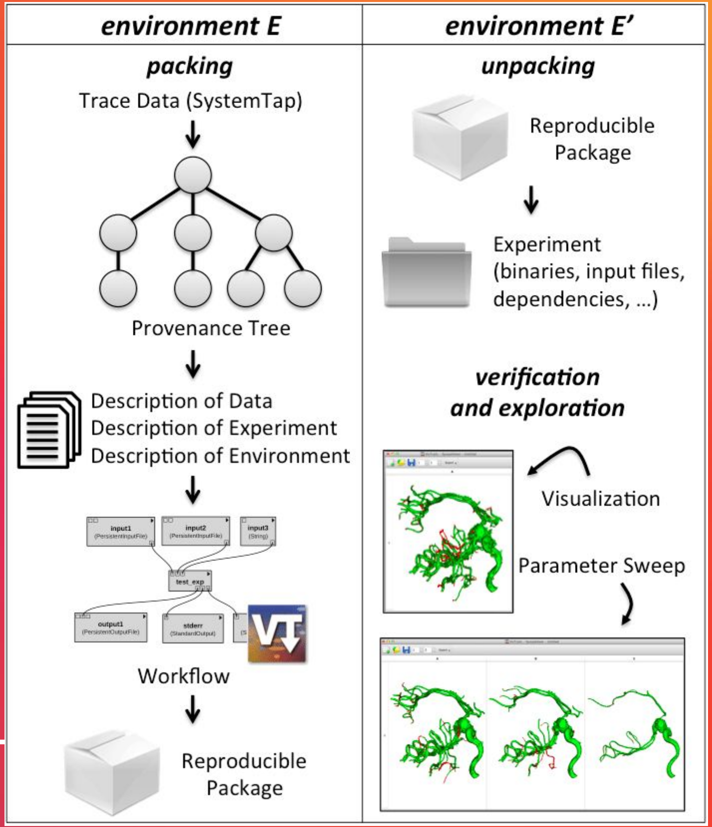


```
AllanTools==2019.9
appnope==0.1.0
attrs==19.3.0
backcall==0.1.0
bleach==3.1.1
certifi==2019.11.28
cycler==0.10.0
decorator==4.4.2
defusedxml==0.6.0
entrypoints==0.3
importlib-metadata==1.5.0
ipykernel==5.1.4
ipython==7.13.0
ipython-genutils==0.2.0
jedi==0.16.0
Jinja2==2.11.1
json5==0.9.0
jsonschema==3.2.0
jupyter-client==6.0.0
jupyter-core==4.6.3
jupyterlab==2.0.1
jupyterlab-server==1.0.7
kiwisolver==1.1.0
MarkupSafe==1.1.1
matplotlib==3.2.0
mistune==0.8.4
nbconvert==5.6.1
nbformat==5.0.4
notebook==6.0.3
numpy==1.18.1
pandas==1.0.2
pandocfilters==1.4.2
parso==0.6.2
```

Slide created by: Tim Head

Improving FAIRness with cutting-edge tools

- ReproZip - Advanced provenance tracking, command recording and encapsulation



environment E

packing

Trace Data (SystemTap)

Provenance Tree

Description of Data
Description of Experiment
Description of Environment

Workflow

Reproducible Package

environment E'

unpacking

Reproducible Package

Experiment (binaries, input files, dependencies, ...)

verification and exploration

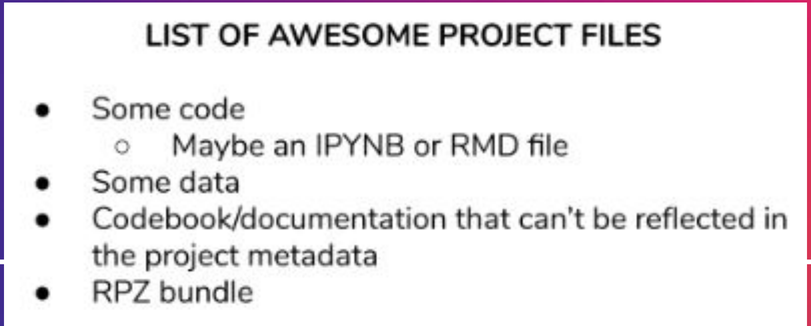
Visualization

Parameter Sweep

input1 (PersistentInputFile)
input2 (PersistentInputFile)
input3 (String)
test_exp
output1 (PersistentOutputFile)
stderr (StandardOutput)
VT

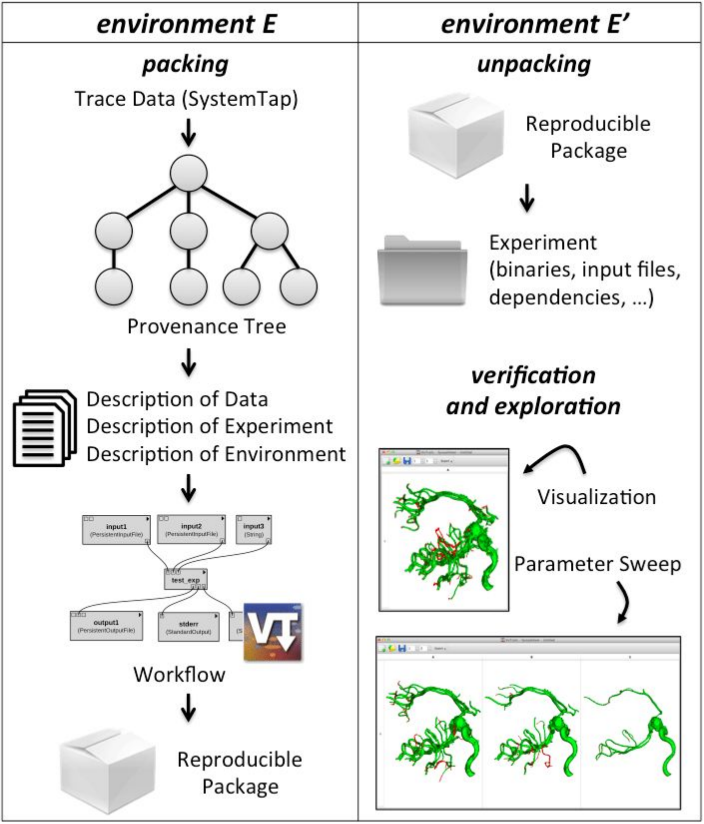
Improving FAIRness with cutting-edge tools

- ReproZip - Advanced provenance tracking, command recording and encapsulation



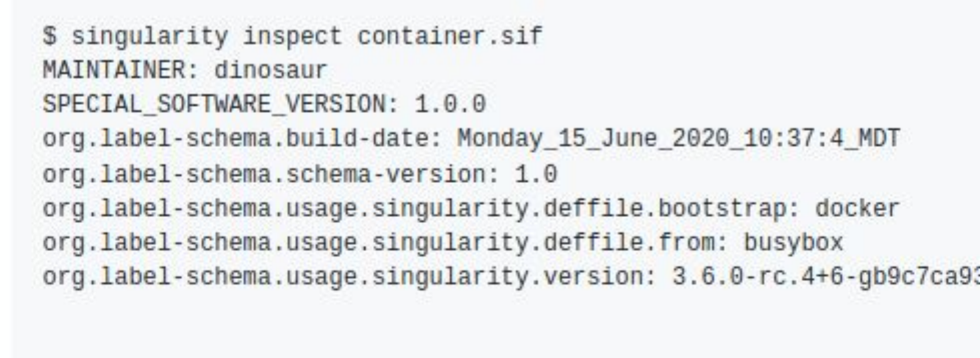
LIST OF AWESOME PROJECT FILES

- Some code
  - Maybe an IPYNB or RMD file
- Some data
- Codebook/documentation that can't be reflected in the project metadata
- RPZ bundle

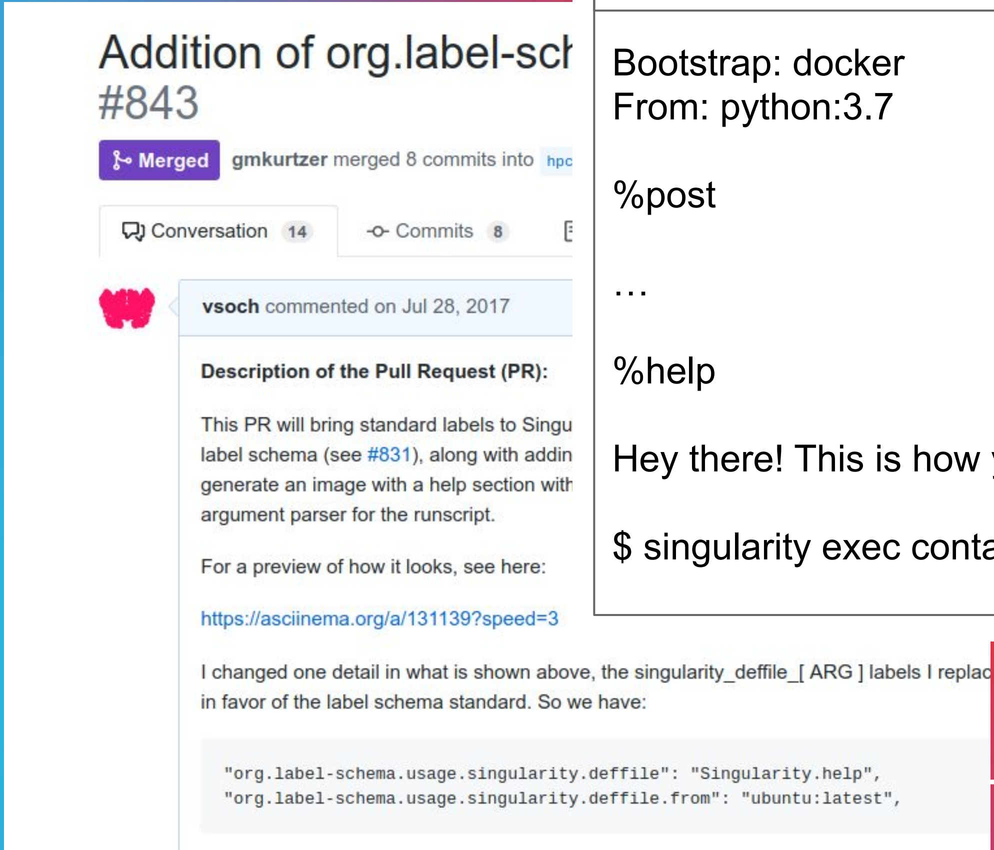


Improving FAIRness with cutting-edge tools

- Singularity - A container technology that supports HPC
- Read-only
- 'Inspect' for metadata and labels



```
$ singularity inspect container.sif
MAINTAINER: dinosaur
SPECIAL_SOFTWARE_VERSION: 1.0.0
org.label-schema.build-date: Monday_15_June_2020_10:37:4_MDT
org.label-schema.schema-version: 1.0
org.label-schema.usage.singularity.deffile.bootstrap: docker
org.label-schema.usage.singularity.deffile.from: busybox
org.label-schema.usage.singularity.version: 3.6.0-rc.4+6-gb9c7ca93
```



Addition of org.label-sch
#843

Merged gmkurtzer merged 8 commits into hpc

Conversation 14 Commits 8

vsoch commented on Jul 28, 2017

Description of the Pull Request (PR):

This PR will bring standard labels to Singu
label schema (see #831), along with addin
generate an image with a help section with
argument parser for the runscript.

For a preview of how it looks, see here:

https://asciinema.org/a/131139?speed=3

I changed one detail in what is shown above, the singularity_deffile_[ARG] labels I replac
in favor of the label schema standard. So we have:

```
"org.label-schema.usage.singularity.deffile": "Singularity.help",
"org.label-schema.usage.singularity.deffile.from": "ubuntu:latest",
```

```
$ singularity inspect -H container.sif
```

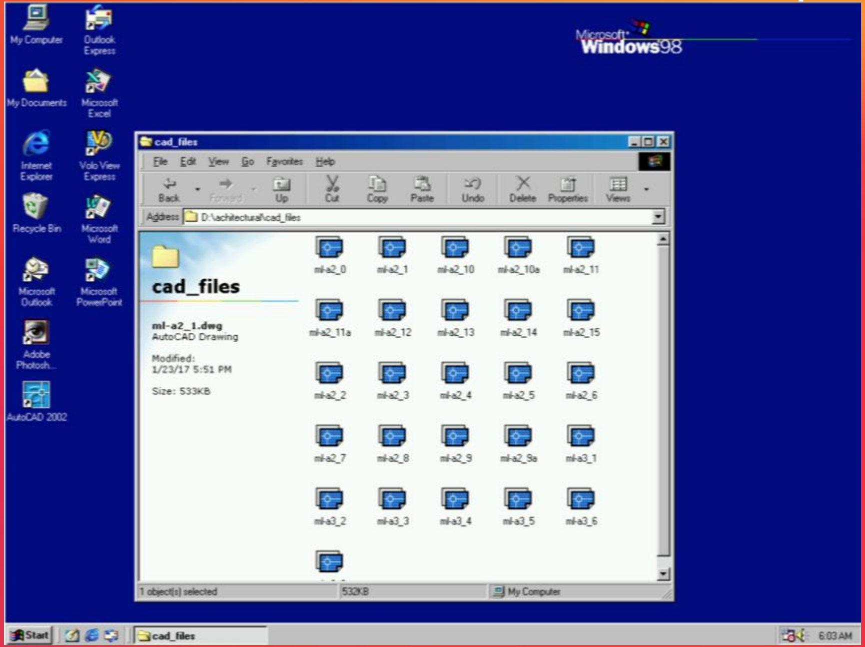
```
Bootstrap: docker
From: python:3.7

%post

…

%help

Hey there! This is how you can run this container:

$ singularity exec container.sif /code/script.py input1
```

Help within container

Slide
created by:
Vanessa
Sochat

Improving FAIRness with cutting-edge tools

- EaaSI - Infrastructure and services for software emulation, sharing, documentation, discovery and access
- Legacy research, support for proprietary software



Address D:\achitectural\cad_files

ml-a2_1.dwg
AutoCAD Drawing

Modified:
1/23/17 5:51 PM

Size: 533KB

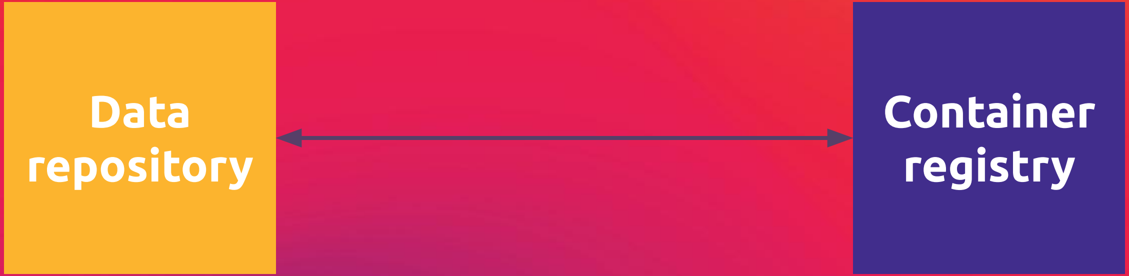
1 object(s) selected | 532KB | My Computer

Halftime summary

- Reproducibility as a problem in science
- Long-term preservation for scientific research
- Great tools that solve these problems
  - FAIR
  - Ease of access for data and code

My view: A good solution for data repositories

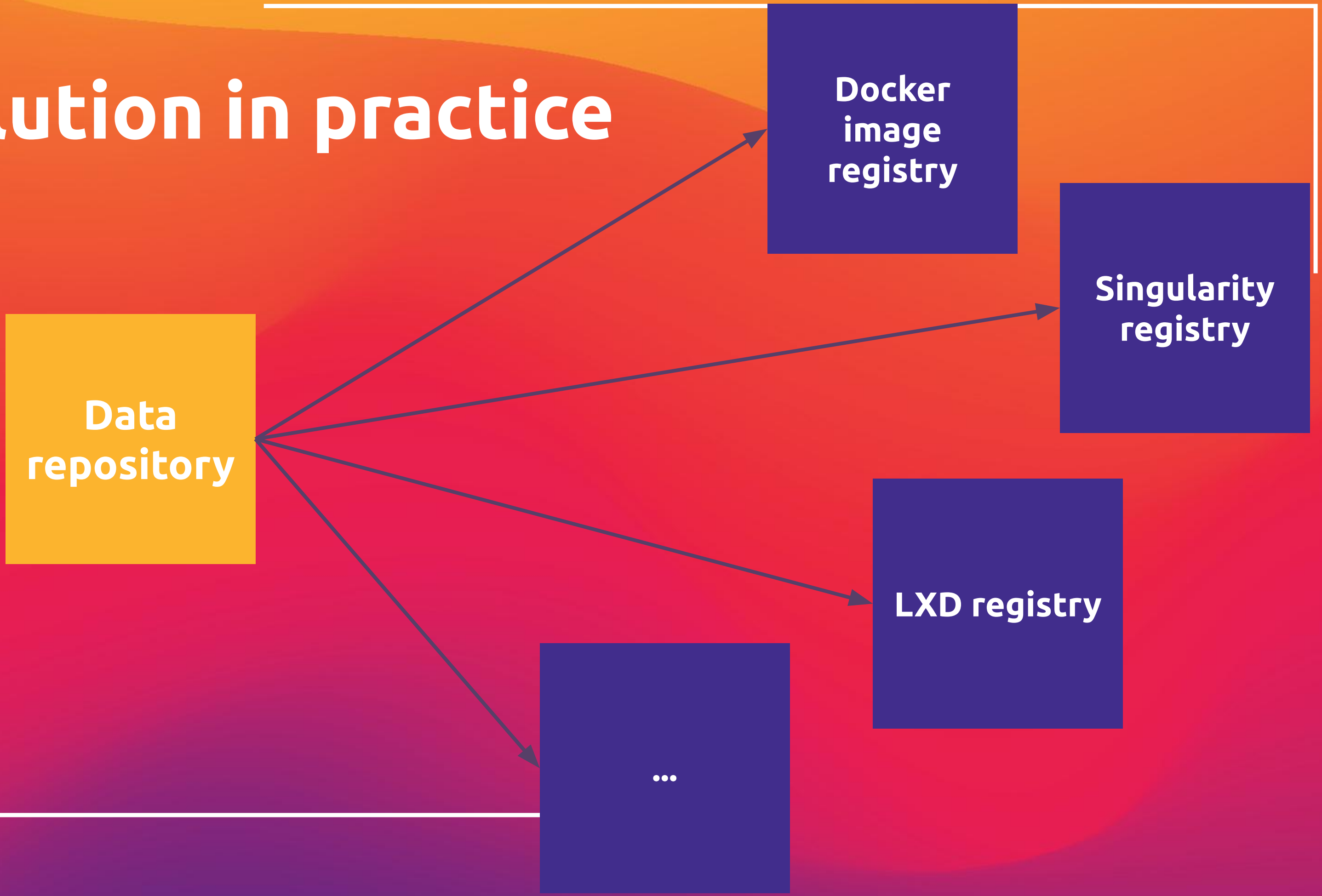
Looking up to software repositories



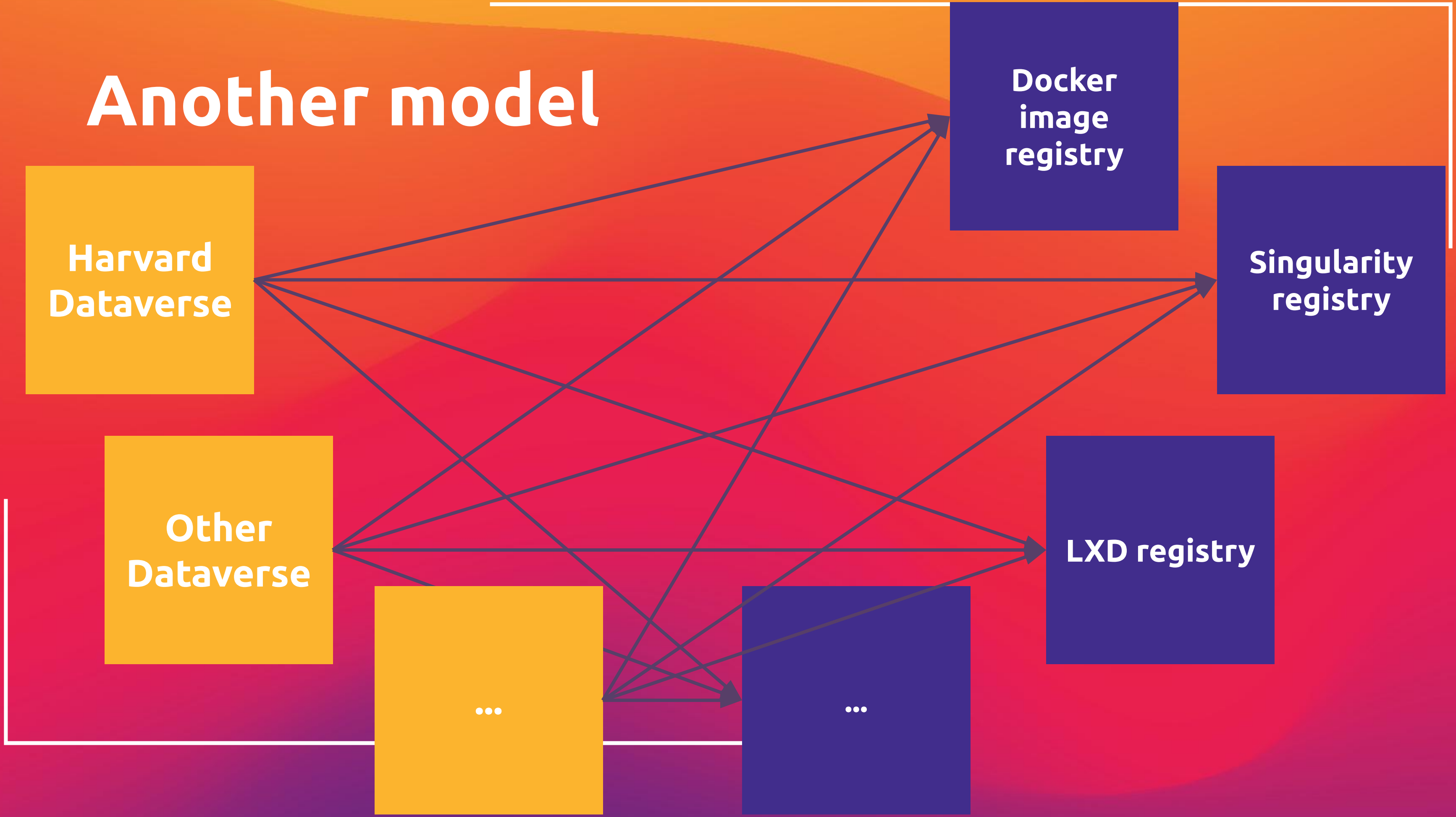
Data repository

Container registry

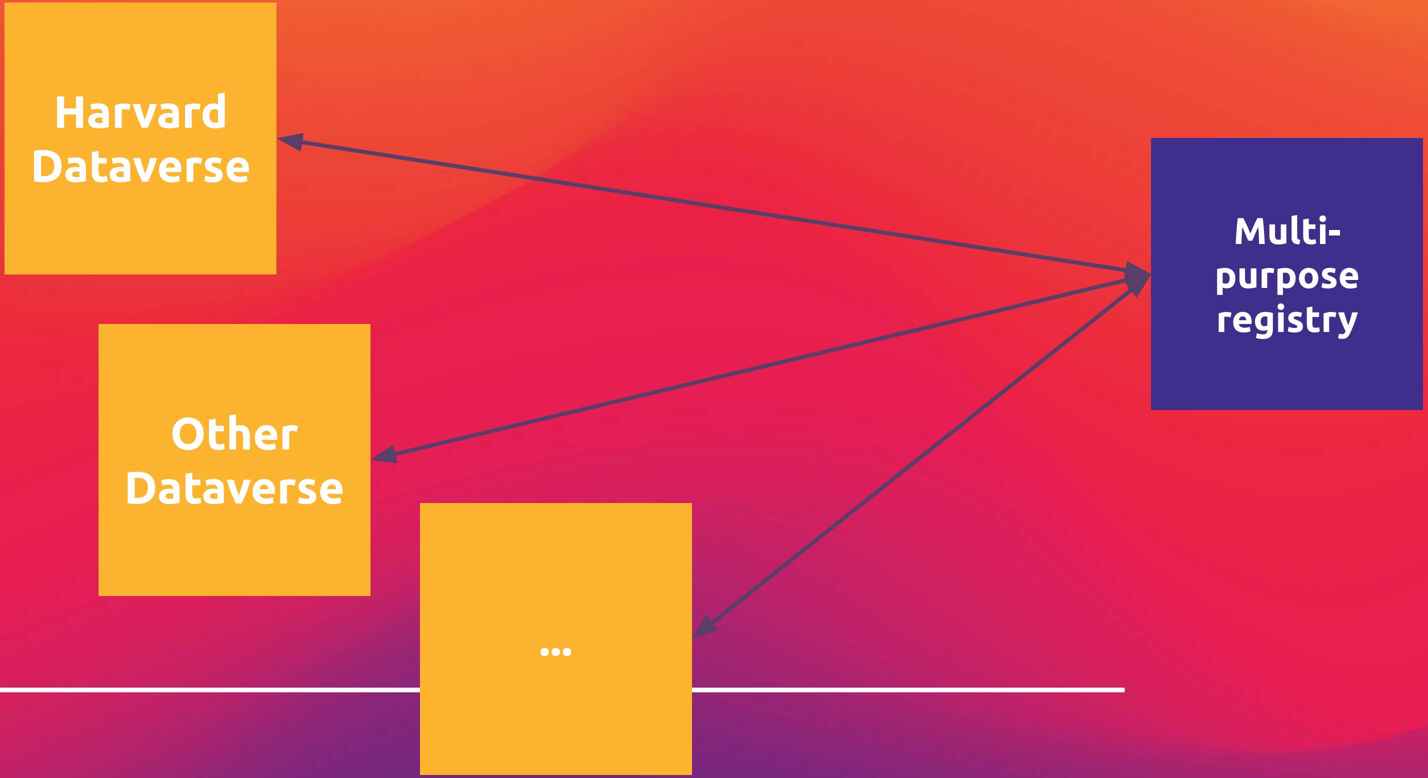
A solution in practice



Another model



A future model



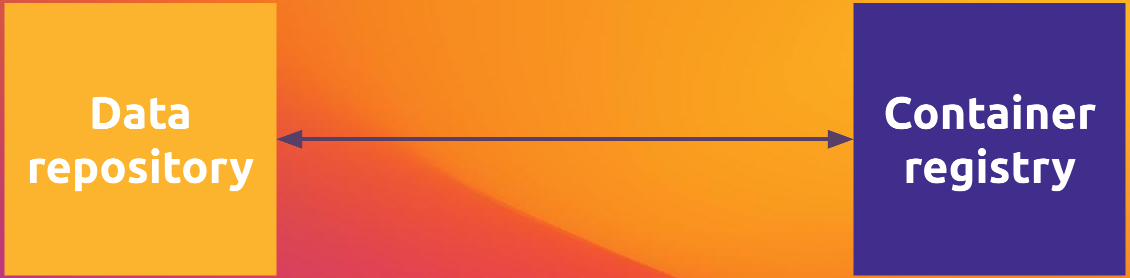
Harvard Dataverse

Other Dataverse

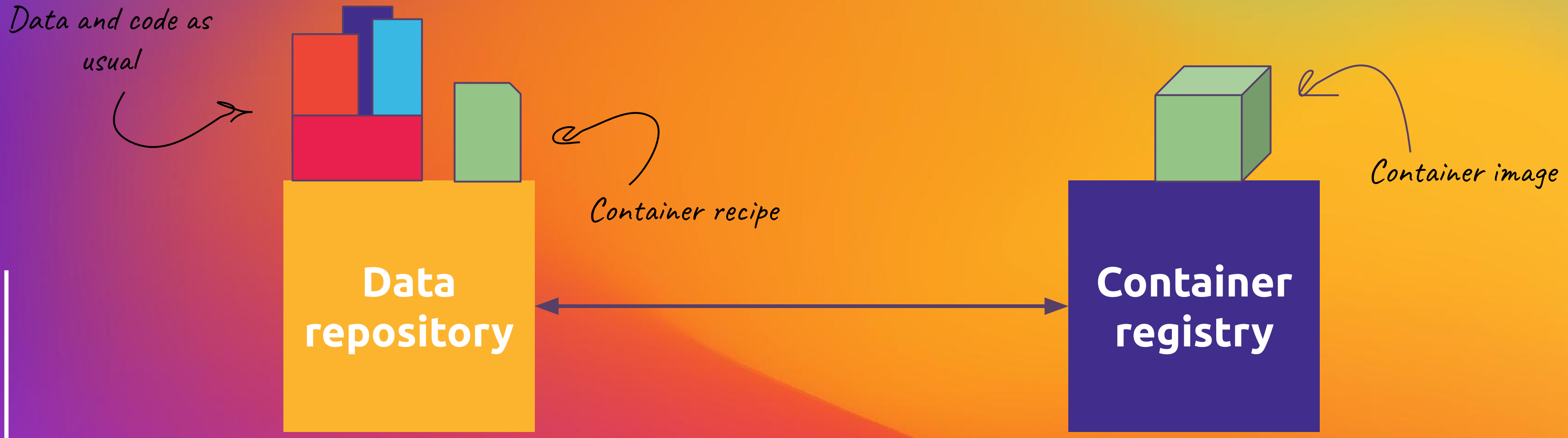
...

Multi-purpose registry

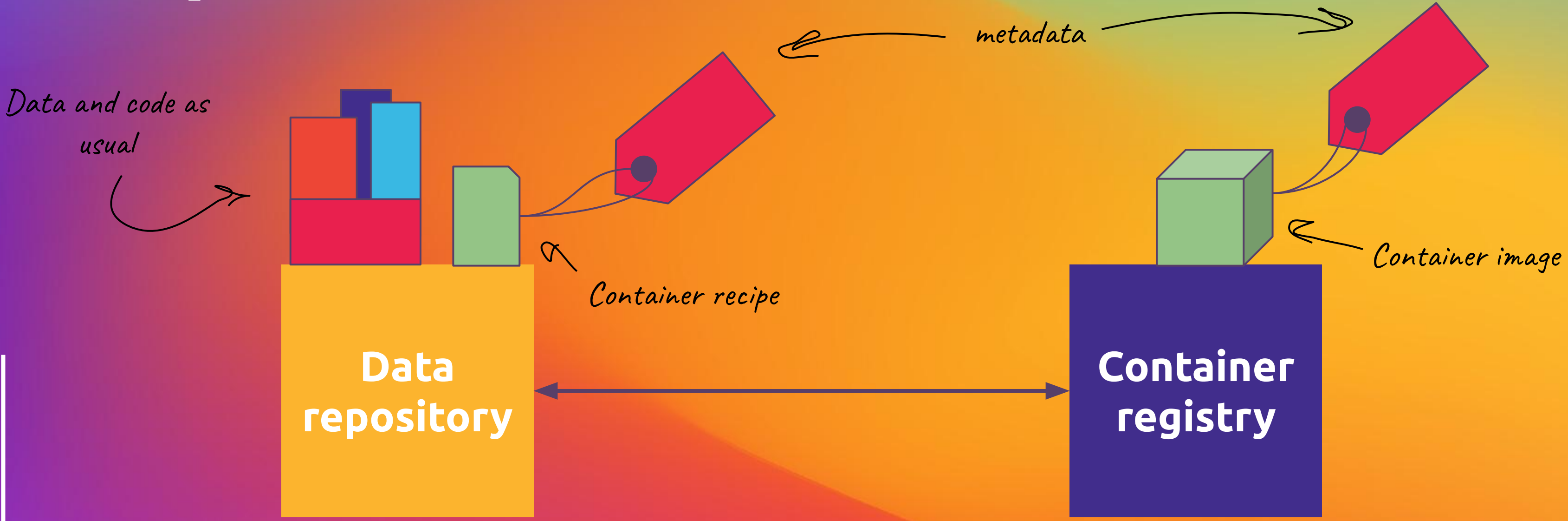
The solution in practice: An implementation



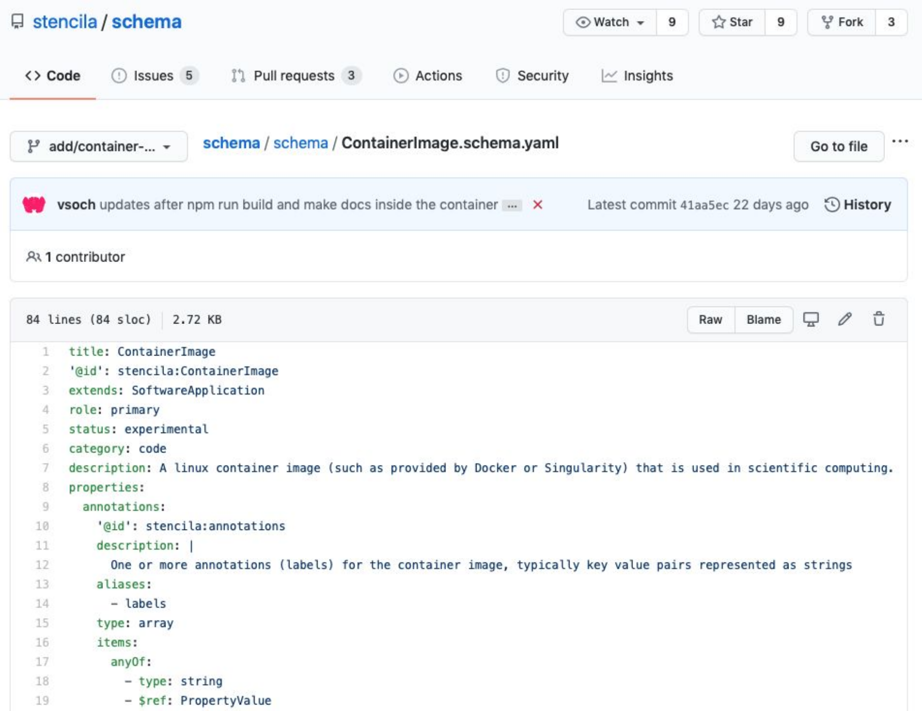
The solution in practice: An implementation



The solution in practice: A FAIR implementation



Metadata for containers currently in development!



schema / schema / ContainerImage.schema.yaml

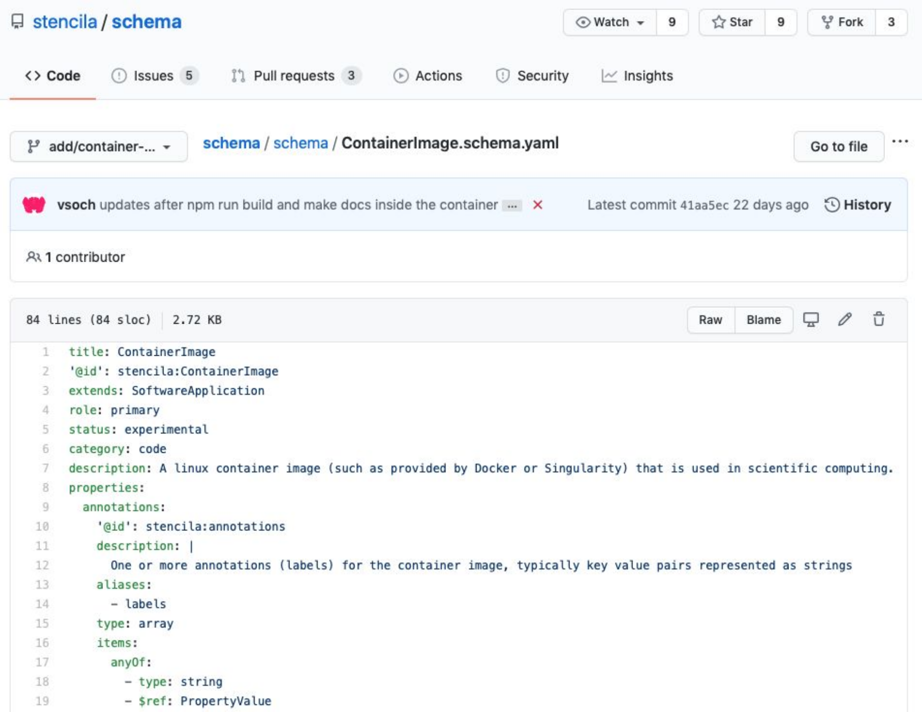
vsoch updates after npm run build and make docs inside the container ... ✕ Latest commit 41aa5ec 22 days ago History

1 contributor

84 lines (84 sloc) | 2.72 KB

```yaml
title: ContainerImage
'@id': stencila:ContainerImage
extends: SoftwareApplication
role: primary
status: experimental
category: code
description: A linux container image (such as provided by Docker or Singularity) that is used in scientific computing.
properties:
  annotations:
    '@id': stencila:annotations
    description: |
      One or more annotations (labels) for the container image, typically key value pairs represented as strings
    aliases:
      - labels
    type: array
    items:
      anyOf:
        - type: string
        - $ref: PropertyValue
```

Metadata for containers currently in development!



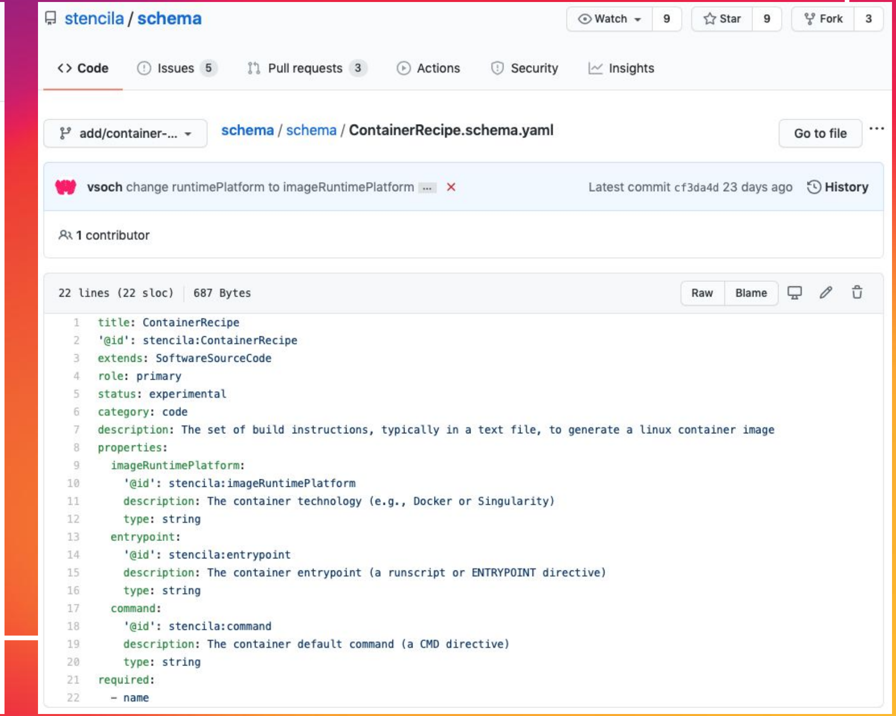
schema / schema / ContainerImage.schema.yaml

vsoch updates after npm run build and make docs inside the container … Latest commit 41aa5ec 22 days ago

1 contributor

84 lines (84 sloc) | 2.72 KB

```yaml
title: ContainerImage
'@id': stencila:ContainerImage
extends: SoftwareApplication
role: primary
status: experimental
category: code
description: A linux container image (such as provided by Docker or Singularity) that is used in scientific computing.
properties:
  annotations:
    '@id': stencila:annotations
    description: |
      One or more annotations (labels) for the container image, typically key value pairs represented as strings
    aliases:
      - labels
    type: array
    items:
      anyOf:
        - type: string
        - $ref: PropertyValue
```



schema / schema / ContainerRecipe.schema.yaml

vsoch change runtimePlatform to imageRuntimePlatform … Latest commit cf3da4d 23 days ago

1 contributor

22 lines (22 sloc) | 687 Bytes

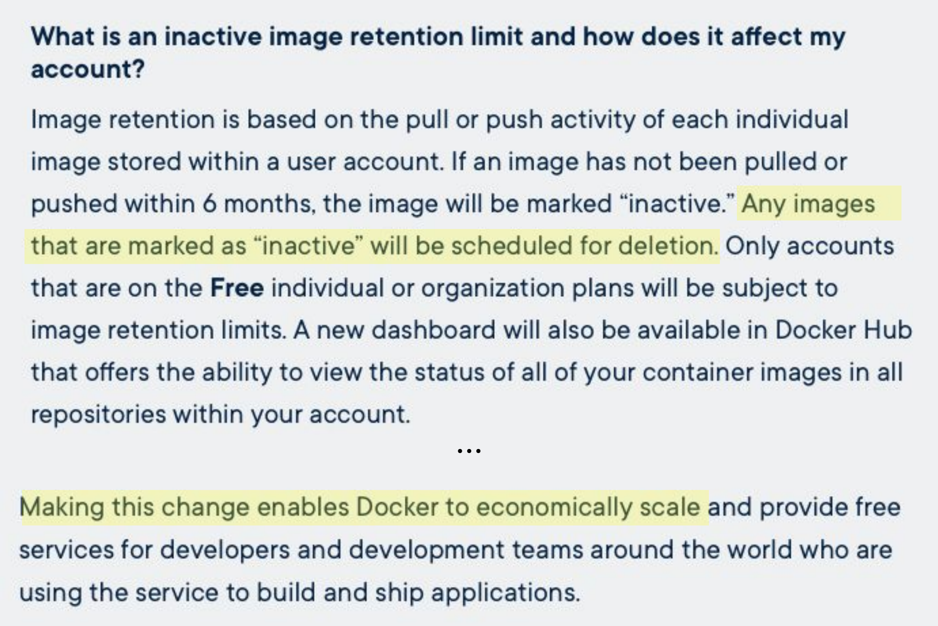
```yaml
title: ContainerRecipe
'@id': stencila:ContainerRecipe
extends: SoftwareSourceCode
role: primary
status: experimental
category: code
description: The set of build instructions, typically in a text file, to generate a linux container image
properties:
  imageRuntimePlatform:
    '@id': stencila:imageRuntimePlatform
    description: The container technology (e.g., Docker or Singularity)
    type: string
  entrypoint:
    '@id': stencila:entrypoint
    description: The container entrypoint (a runscript or ENTRYPOINT directive)
    type: string
  command:
    '@id': stencila:command
    description: The container default command (a CMD directive)
    type: string
required:
  - name
```

Good outcomes:

- Potential to vastly improve reproducibility and reusability for small(er)-scale studies
  - Not too late to encapsulate old code!
- Data repositories would support research dissemination for different computing infrastructures (cloud or HPC with Singularity)
- Easy integration with most reproducibility tools

Caveats

- While data repositories easily support multiple metadata standards, setting up a container registry may be more complicated and expensive



What is an inactive image retention limit and how does it affect my account?

Image retention is based on the pull or push activity of each individual image stored within a user account. If an image has not been pulled or pushed within 6 months, the image will be marked “inactive.” Any images that are marked as “inactive” will be scheduled for deletion. Only accounts that are on the **Free** individual or organization plans will be subject to image retention limits. A new dashboard will also be available in Docker Hub that offers the ability to view the status of all of your container images in all repositories within your account.

…

Making this change enables Docker to economically scale and provide free services for developers and development teams around the world who are using the service to build and ship applications.

Potential solutions

- Standardized containers for repository users
  - Same base layers
- Containers generated by user-friendly reproducibility platforms
- Proprietary containers treated as sensitive data

Conclusion

- Code on data repositories creates need to adequately support it
- Many options are possible and FAIR
- Investing in container registry would be the best long-term solution

Thank you for your attention!

ALFRED P. SLOAN
FOUNDATION