FAIR Assessment for Research Software

deRSE25, Karlsruhe

Elena Breitmoser, e.breitmoser@epcc.ed.ac.uk
Faruk Diblen, f.diblen@esciencecenter.nl

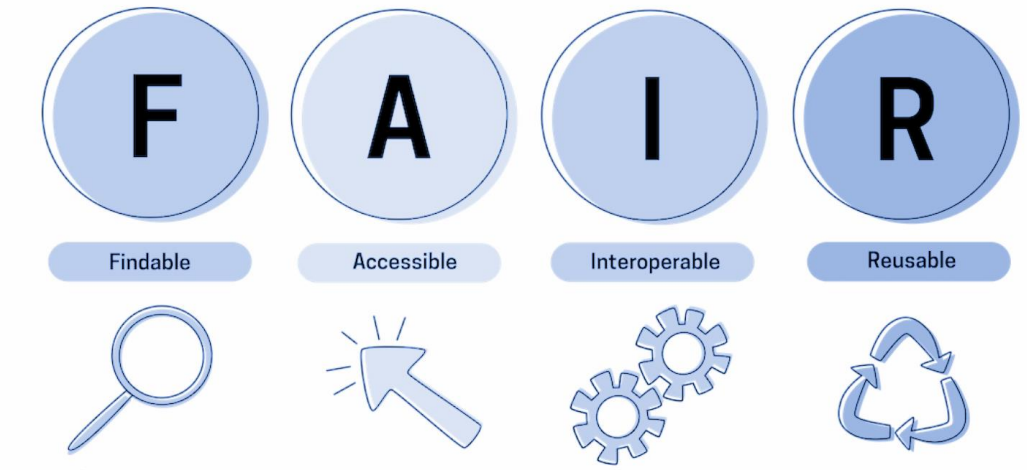
Funded by the European Union

Understanding FAIR principles for research software

Quick review of FAIR principles

- Principles provide guidance & are non-normative statements
  - No prescriptive language, i.e. no MUSTs/SHOULDs/MAYs…
- Widely accepted and praised
  - Facilitate provenance
  - Allows credit to be given
    - For you or to those that produce the resources you use
  - Other efforts build on FAIR
    - Open Science
    - Reproducibility, etc
- Started in 2016 for data (Wilkinson *et al.)*
- FAIR principles are now devised for other digital objects beyond data
  - Research software (2021 Chue Hong *et al* : FAIR for Research Software Principles (FAIR4RS)),…

Many principles are shared between FAIR data & sw



FAIR4RS Principles v1.0

F: Software, and its associated metadata, is easy for both humans and machines to find

F1. Software is assigned a globally unique and persistent identifier.
- F1.1. Components of the software representing levels of granularity are assigned distinct identifiers.
- F1.2. Different versions of the software are assigned distinct identifiers.

F2. Software is described with rich metadata.
F3. Metadata clearly and explicitly include the identifier of the software they describe.
F4. Metadata are FAIR, searchable and indexable.

A: Software, and its metadata, is retrievable via standardized protocols.

A1. Software is retrievable by its identifier using a standardized communications protocol.
- A1.1. The protocol is open, free, and universally implementable.
- A1.2. The protocol allows for an authentication and authorization procedure, where necessary.

A2. Metadata are accessible, even when the software is no longer available.

I: Software interoperates with other software by exchanging data and/or metadata, and/or through interaction via application programming interfaces (APIs), described through standards.

I1. Software reads, writes and exchanges data in a way that meets domain-relevant community standards.
I2. Software includes qualified references to other objects

R: Software is both usable (can be executed) and reusable (can be understood, modified, built upon, or incorporated into other software).

R1. Software is described with a plurality of accurate and relevant attributes.
- R1.1. Software is given a clear and accessible license.
- R1.2. Software is associated with detailed provenance.

R2. Software includes qualified references to other software.
R3. Software meets domain-relevant community standards.

Chue Hong, N. P., et al. (2022). FAIR Principles for Research Software version 1.0. (FAIR4RS Principles v1.0). Research Data Alliance. DOI: https://doi.org/10.15497/RDA00068

How do we measure the FAIRness of RS?

Metrics – a good metric should be (fairmetrics.org):

1. **Clear**:
   - anyone can understand purpose of the metric
2. **Realistic**:
   - should not be unduly complicated for a resource to comply with the metric
3. **Discriminating**:
   - metric should measure something important for FAIRness
   - distinguish the degree to which that resource meets that objective
   - be able to provide instruction as to what would maximize that value
4. **Measurable**:
   - assessment can be made in an objective, quantitative, machine-interpretable, scalable and reproducible manner
   - ensuring transparency of what is being measured & how
5. **Universal**:
   - the metric should be applicable to all digital resources

FAIR Research Software metrics

Outcomes of the FAIR-IMPACT project –

D5.2 - Metrics for automated FAIR software assessment in a disciplinary context.

DOI 10.5281/zenodo.10047401

Identifier	Name
FRSM-01	Does the software have a globally unique and persistent identifier?
FRSM-02	Do the different components of the software have their own identifiers?
FRSM-03	Does each version of the software have a unique identifier?
FRSM-04	Does the software include descriptive metadata which helps define its purpose?
FRSM-05	Does the software include development metadata which helps define its status?
FRSM-06	Does the software include metadata about the contributors and their roles?
FRSM-07	Does the software metadata include the identifier for the software?
FRSM-08	Does the software have a publicly available, openly accessible and persistent metadata record?
FRSM-09	Is the software developed in a code repository / forge that uses standard communications protocols?

Identifier	Name
FRSM-10	Are the formats used by the data consumed or produced by the software open and a reference provided to the format?
FRSM-11	Does the software use open APIs that support machine-readable interface definition?
FRSM-12	Does the software provide references to other objects that support its use?
FRSM-13	Does the software describe what is required to use it?
FRSM-14	Does the software come with test cases to demonstrate it is working?
FRSM-15	Does the software source code include licensing information for the software and any bundled external software?
FRSM-16	Does the software metadata record include licensing information?
FRSM-17	Does the software include provenance information that describe the development of the software?

How to promote FAIRness

How FAIR are you?

- How can you determine your FAIR level?
  - Do it yourself - requires knowledge
  - Guided self-assessments - requires interpretation by you
  - Automated assessments - application does it for you (with caveats)
  - How suitable are existing **automated** tools for assessing FAIRness, when applied to software?
- What is your current FAIRness baseline and how can you improve?

Overview of assessment approaches and tools

Guided approaches

Asking a series of questions to a human

You assess yourself as to whether your resources satisfy FAIR principles, e.g.,

1. https://fair-software.nl/ (**FAIR Software** but does not explicitly use FAIR4RS)
2. https://satifyd.dans.knaw.nl/ (for FAIR data)
3. https://ardc.edu.au/resource/fair-data-self-assessment-tool/ (FAIR data)
4. https://fairsoftwarechecklist.net/v0.2/ (**FAIR software**, inspired by ARDC's FAIR data self-assessment tool and by the outcomes of the FAIR4RS Working Group

Overview of assessment approaches and tools

Automated approaches

Interrogating the software using a combination of machine-actionable tests (more consistent, objective, quicker)

Use a web service or application that generates your FAIRness level (score/badge)

Tooling to assess FAIRness of Research Software (RS) not as mature as for FAIR Data

Comparison of tools for automated FAIR software assessment DOI 10.5281/zenodo.13268685

1. **F-UJI** - https://www.f-uji.net/ (Web, Data (mostly)) – we have been working on extension for Research Software for it
2. **Howfairis** - https://github.com/fair-software/howfairis (Python app, RS but only for GitHub/GitLab (not self-hosted))
3. FAIR-Checker - https://fair-checker.france-bioinformatique.fr/ (Web, Data)
4. FAIR-Enough - https://fair-enough.semanticscience.org/ (Web, Data)
5. OpenEBench - https://openebench.readthedocs.io/en/latest/

How to promote FAIRness

Importance of early & continuous integration of FAIRness during sw development process

- Provision of guidelines, processes, tools
- Examples – easy to replicate & extend to other sw projects

Avoid duplication to simplify consistent maintenance

- E.g., info kept in README, codemeta or CITATION CFF file

Definite interest & need for automated FAIR assessment tools

- Improve F-UJI tool for Research Software – implement more tests
- But: human-readability needs to be maintained!

Need for transparency & precise guidelines:

- What exactly is/not measured?
- Why does my repo fail for a given test – what can I do to improve it (quickly)?
- Why do I get different scores for very similar repos?

How to promote FAIRness

With automated assessment tools

Raise awareness of tools/practices that cover FAIRness with little effort for sw developers/researchers:

- Repository frameworks such as github have already tools in place
  - Generate a list of authors
  - Code contributors
  - Coding languages used
  - Licence provision
  - Etc.
- 50% of metrics easily satisfied:
  - Use general-purpose, open repo Zenodo with github integration:
    - Authors can be credited easily
    - Adds DOI
  - Use machine-readable files
  - Improve README
  - Add codemeta file
- List main hands-on tools that help to generate metadata files or badges automatically
  - To create codemeta.json, CITATION.CFF, development status badge,…

F-UJI: Automated assessment tool for the FAIRness of RS

Extension of F-UJI to Research Software

Existing automated assessment tool for data: F-UJI https://www.f-uji.net/

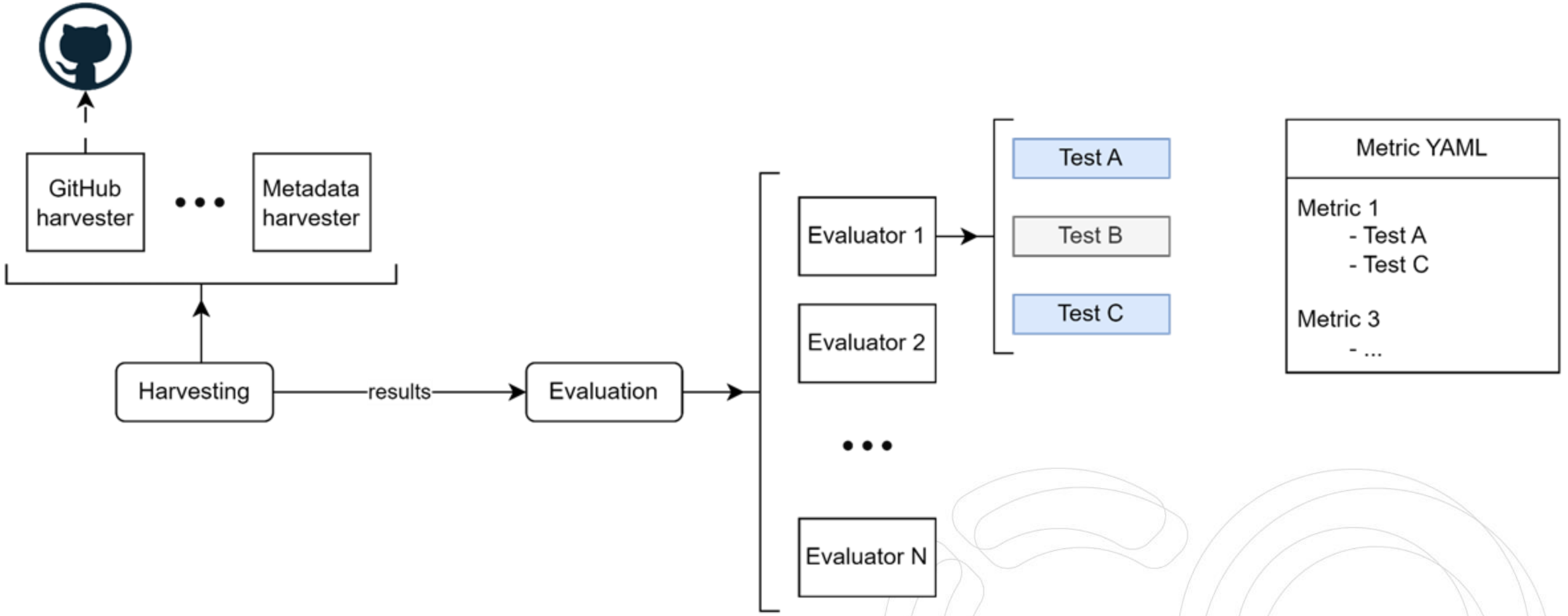
- Next version release will include our changes for Research Software, available through their web client

F-UJI extension for Research Software (POC): https://github.com/softwaresaved/fuji/

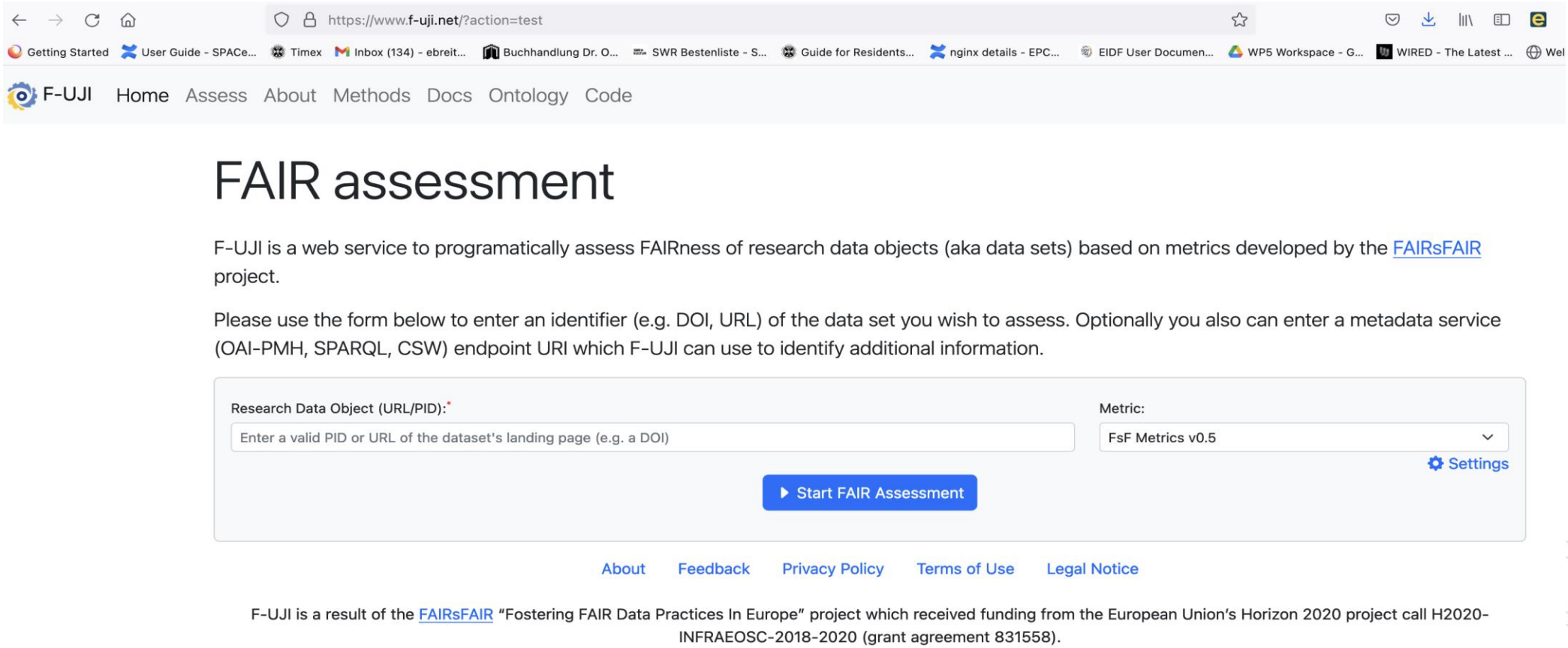
- Merged back into original F-UJI repo
- Not all metrics have been implemented yet
  - General, agnostic test implementations
  - Domain-specific test implementations
- M5.6 Practical tests for automated FAIR software assessment in a disciplinary context

DOI 10.5281/zenodo.10890043

How does F-UJI evaluate Research Software



F-UJI demo



This work was supported by the Edinburgh International Data Facility (EIDF) and the Data-Driven Innovation Programme at the University of Edinburgh.

F-UJI demo

Metric Specification:	https://doi.org/10.5281/zenodo.6461229
Software version:	3.2.0

Summary:

8.89%

	Score earned:	Fair level:
Findable:	0 of 20	incomplete
Accessible:	0 of 2	incomplete
Interoperable:	0 of 7	incomplete
Reusable:	4 of 16	initial

Report:

Findable

FRSM-01-F1 - Does the software have a globally unique and persistent identifier?

FRSM-02-F1.1 - Do the different components of the software have their own identifiers?

FRSM-03-F1.2 - Does each version of the software have a unique identifier?

FRSM-04-F2 - Does the software include descriptive metadata which helps define its purpose?

F-UJI demo

FRSM-15-R1.1 - The software source code includes licensing information for the software and any bundled external software.

FAIR level: 3 of 3 advanced

Score: 2 of 3

Output:


```
[
    {
        "license": "Apache License 2.0",
        "osi_approved": true,
        "details_url": "http:\/\/spdx.org\/licenses\/Apache-2.0.html"
    }
]
```


Metric tests:

Test:	Test name:	Score:	Maturity:	Result:
FRSM-15-R1.1-1	License file is included.	1	1	
FRSM-15-R1.1-2	The source code includes licensing information for all components bundled with that software.			
FRSM-15-R1.1-3	Recognized licence is in SPDX format.	1	3	

Debug messages:

Level:	Message:
INFO	License verification name through SPDX registry -: Apache License 2.0
INFO	Found SPDX license representation -: http://spdx.org/licenses/Apache-2.0.json
SUCCESS	Found SPDX license representation (spdx url, osi_approved)
SUCCESS	Found licence file: ['LICENSE'].
INFO	Will consider all SPDX licenses as community specific licenses for FRSM-15-R1.1
INFO	This test is not defined in the metric YAML and therefore not performed: FRSM-15-R1.1-CESSDA-1
WARNING	Test for license information of bundled components is not implemented (FRSM-15-R1.1-2).
INFO	This test is not defined in the metric YAML and therefore not performed: FRSM-15-R1.1-CESSDA-3
INFO	This test is not defined in the metric YAML and therefore not performed: FRSM-15-R1.1-CESSDA-2

FRSM-16-R1.1 - Does the software metadata record include licensing information?

F-UJI demo: Not-yet implemented tests

FRSM-13-R1 - Does the software describe what is required to use it?

FRSM-14-R1 - Does the software come with test cases to demonstrate it is working?

FAIR level: 0 of 3 incomplete
Score: 0 of 3
Output: []

Metric tests:

Test:	Test name:	Score:	Maturity:	Result:
FRSM-14-R1-1	Tests and data are provided to check that the software is operating as expected.			
FRSM-14-R1-2	Automated unit and system tests are provided.			
FRSM-14-R1-3	Code coverage / test coverage is reported.			

Debug messages:

Level:	Message:
WARNING	Test for presence of tests and test data is not implemented.
WARNING	Test for Automated unit and system tests is not implemented.
WARNING	Test for code coverage is not implemented.
INFO	This test is not defined in the metric YAML and therefore not performed: FRSM-14-R1-CESSDA-1
INFO	This test is not defined in the metric YAML and therefore not performed: FRSM-14-R1-CESSDA-2
INFO	This test is not defined in the metric YAML and therefore not performed: FRSM-14-R1-CESSDA-3
WARNING	Failed to check the software version identifier.

FRSM-15-R1.1 - The software source code includes licensing information for the software and any bundled external software.

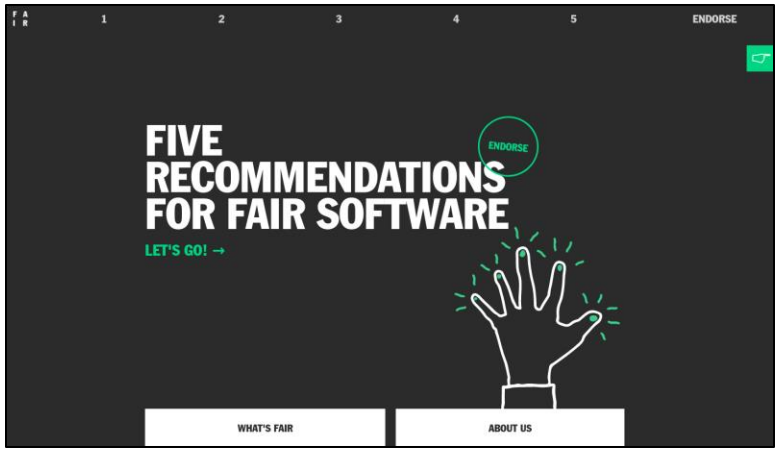
Try it yourself:

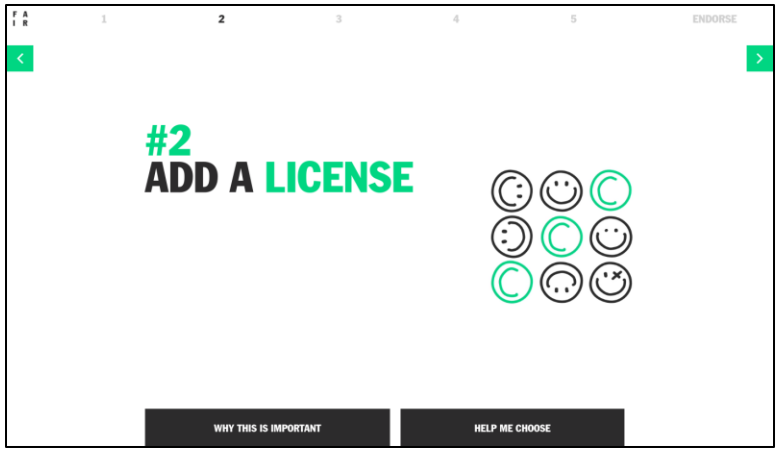
turnip.eidf.ac.uk

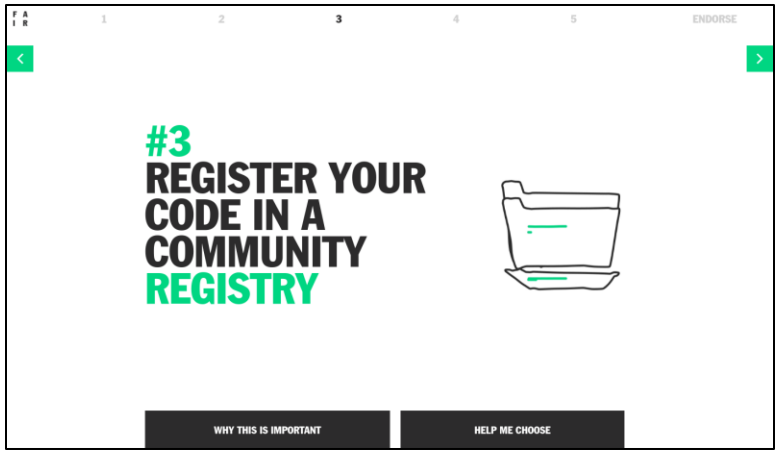
Example git repos:

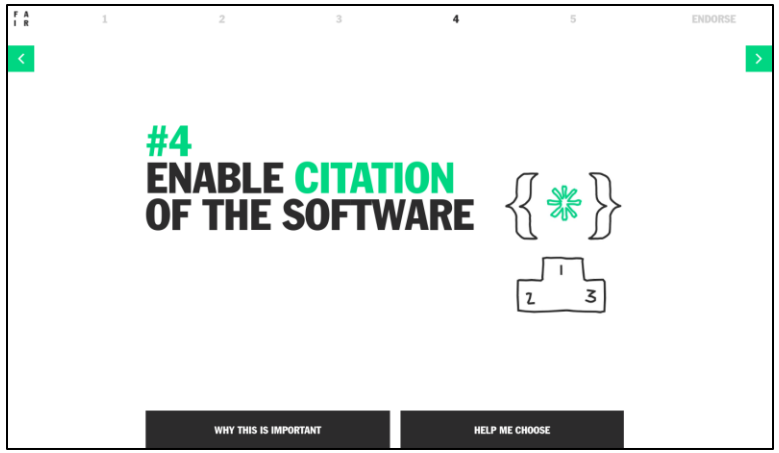
- https://www.f-uji.net/

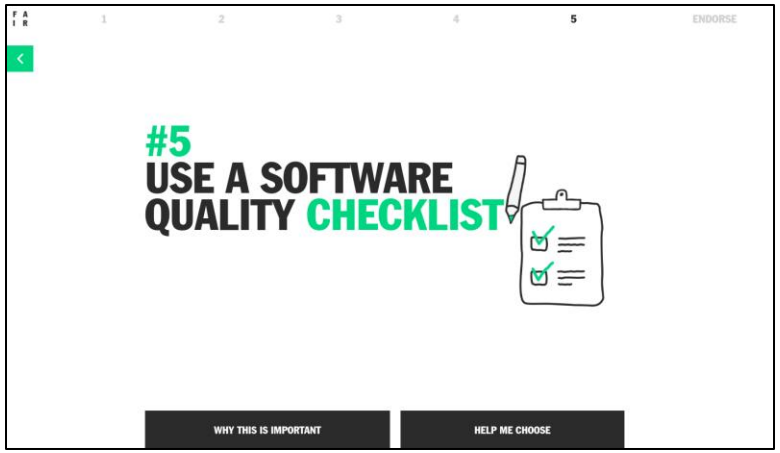
Five recommendations





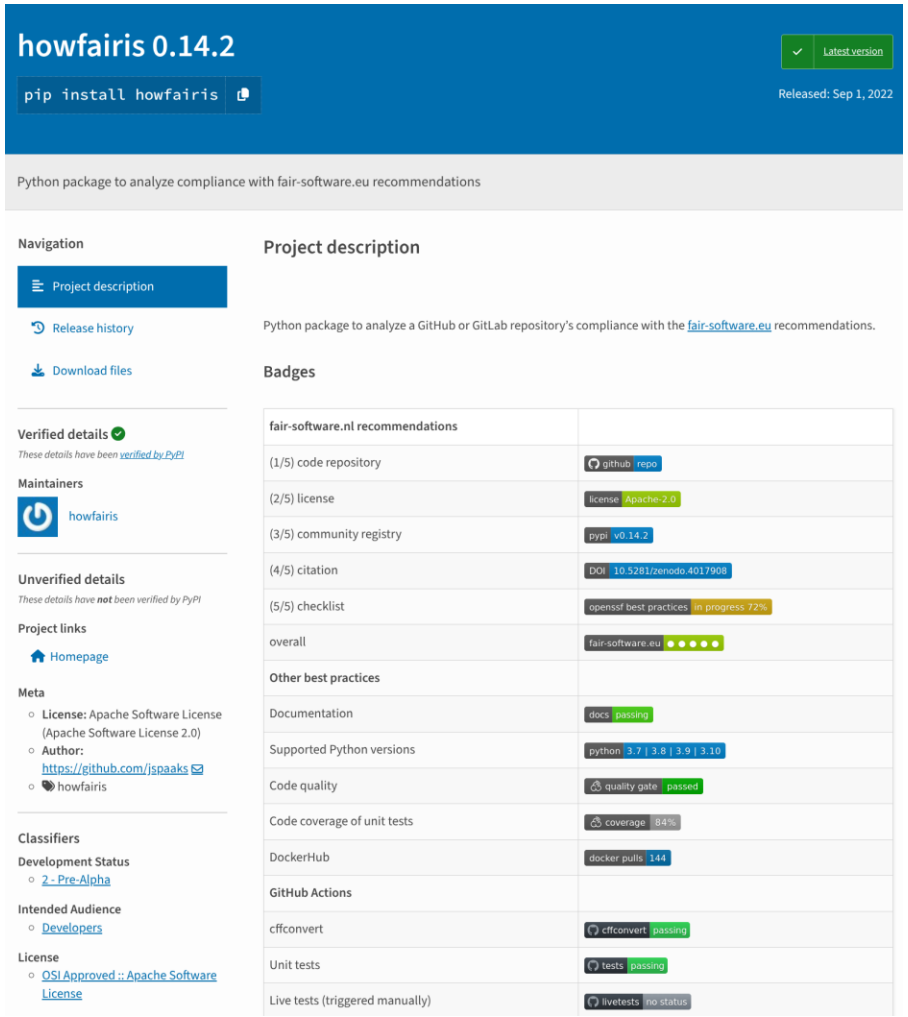






In 2019, we created **https://fair-software.eu** with
5 practical recommendations on how to make your software FAIR

How can we check the compliance automatically?



https://pypi.org/project/howfairis

https://github.com/fair-software/howfairis

Installation

```
pip3 install --user howfairis
```

Usage

```
howfairis https://github.com/<owner>/<repo>
```

`howfairis` supports URLs from the following code repository platforms:

1. `https://github.com`
2. `https://gitlab.com` (not including self-hosted instances)

Badges

~400 badges on GitHub

Gitlab? (self-hosted instances)

https://doi.org/10.5281/zenodo.7193991

18

https://github.com/fair-software/howfairis-github-action

Assess compliance with fair-software.eu

To enable this checker, add the following snippet as `.github/workflows/fair-software.yml` in your GitHub repository.

```yaml
name: fair-software

on: push

jobs:
  verify:
    name: "fair-software"
    runs-on: ubuntu-latest
    steps:
      - uses: fair-software/howfairis-github-action@0.2.1
        name: Measure compliance with fair-software.eu recommendations
        env:
          PYCHARM_HOSTED: "Trick colorama into displaying colored output"
        with:
          MY_REPO_URL: "https://github.com/${{ github.repository }}"
```

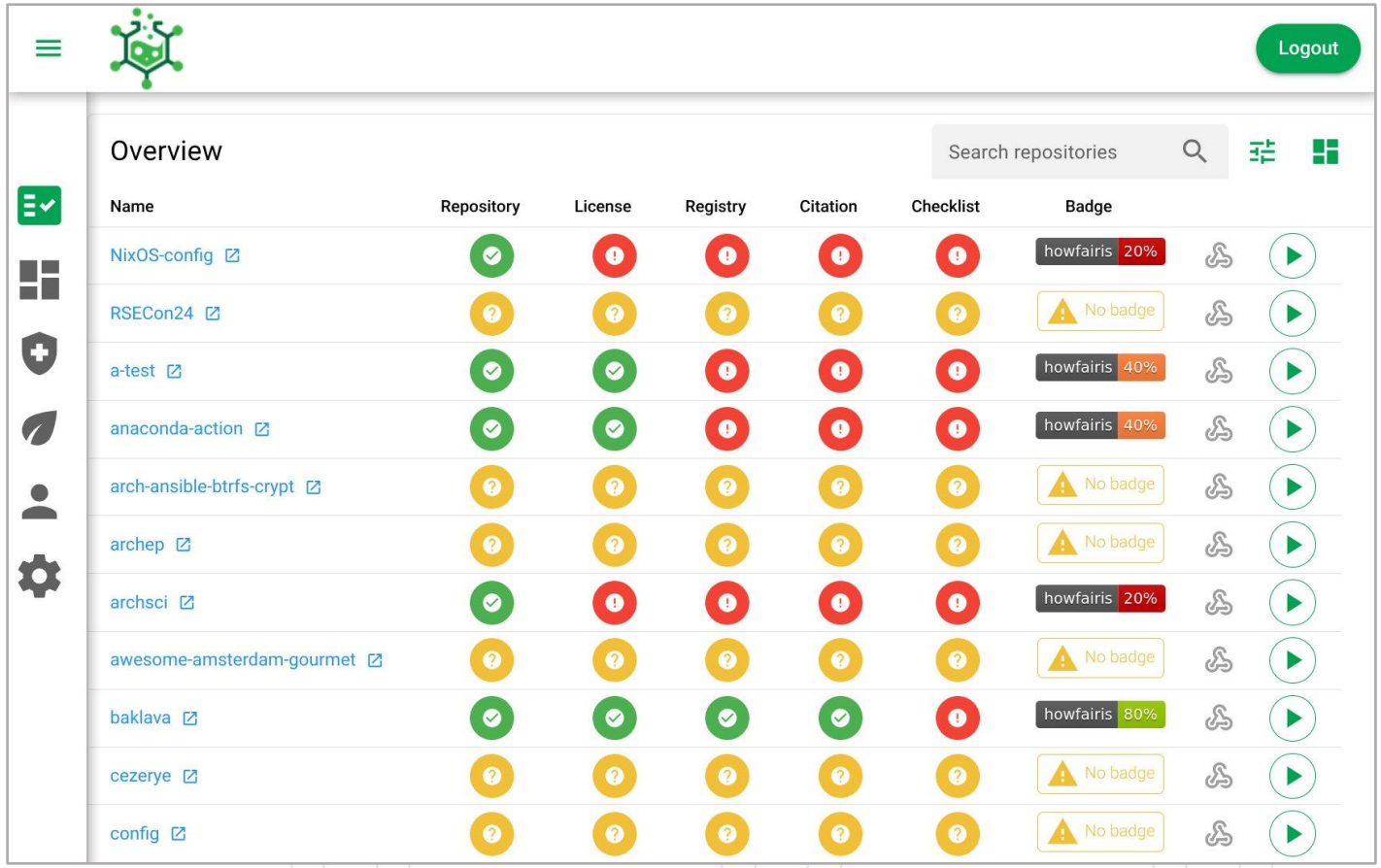
https://doi.org/10.5281/zenodo.7193991

Howfairis as a service

- Cloud service to check compliance using howfairis
- No need to install any tools
- Overview of the compliance
- Interactive dashboard (WIP)
- Extra metrics (e.g. community health)

How can you get involved/help?:

- Test users
- Collaborations
- Funding



https://www.howfairis.com/
https://app.howfairis.com/

Online demo! You can also try it yourself...

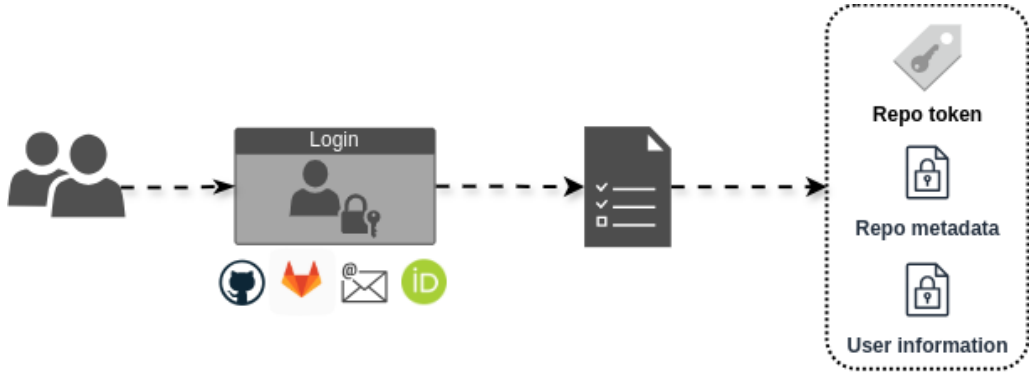
EVERSE Quality Dashboard

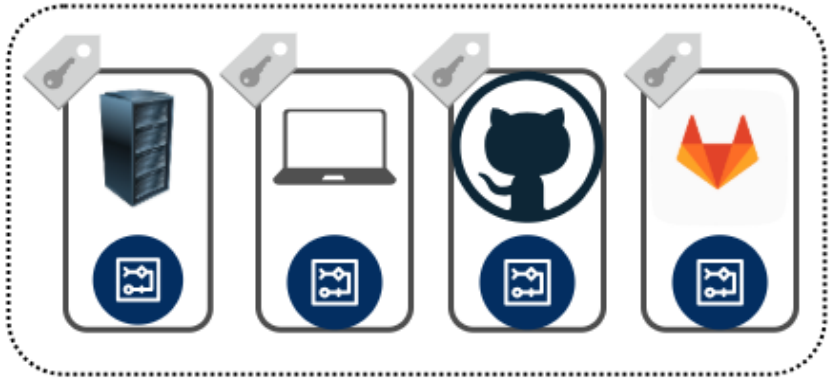
EVERSE: European Virtual Institute for Research Software Excellence

The EVERSE project aims to create a framework for research software and code excellence, **collaboratively designed and championed by the research communities**, in pursuit of building a European network of Research Software Quality and setting the foundations of a future Virtual Institute for Research Software Excellence.

Some of the main goals

- Defining the best practices for research software quality
- Community building
- Training
- Designing pipelines and workflows **using existing tools and services for research software quality assessment**
- Development of a dashboard to display assessment results





Join us!

Any individual or organisation that agrees with our vision statement is welcome to join the network.

https://everse.software/network/

21

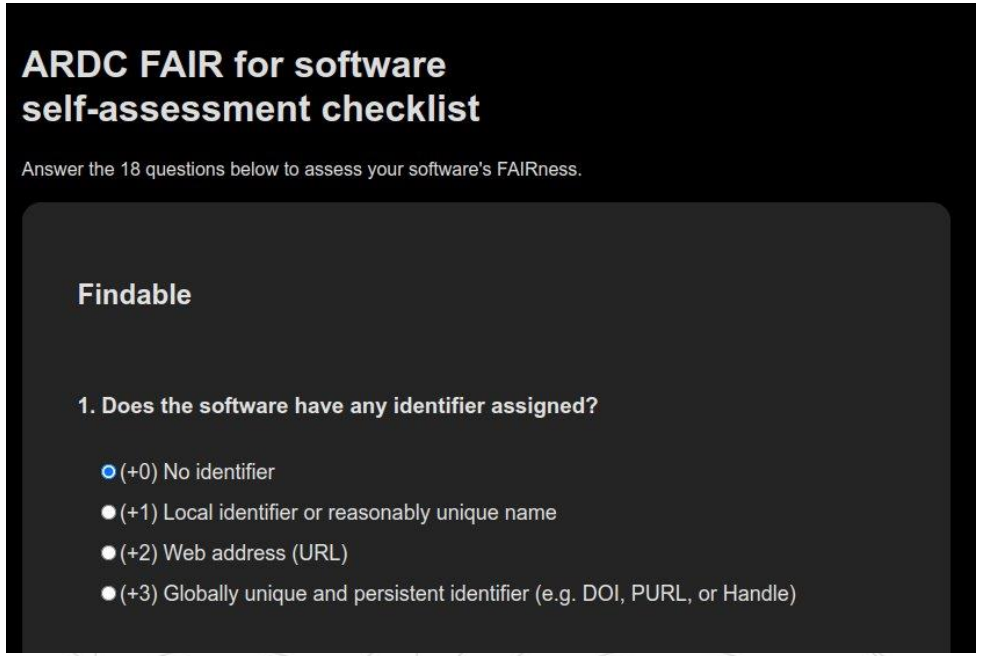
Summary & Conclusions

- Any completely automated tools to assess FAIRness according to FAIR4RS principles?
- Challenges to implement some of the metrics into automated tools at all!

- Do you know of or work on similar tools?
- Questions?

Backups

ARDC FAIR-software checklist



https://doi.org/10.5281/zenodo.7193991

eosc | EVERSE

EVERSE Dashboard

- Authentication
- Assessment
- Dashboard

EVERSE System Design