The Journal of Open Source Software

FACILE-RS: archival and long term preservation of research software repositories made easy

Jochen Klar [1], **Marie Houillon** [2], **Axel Loewe** [2], **Ziad Boutanios**[2], **and Tomas Stary** [2]

1 Independent Software Developer, Germany **2** Karlsruhe Institute of Technology, Germany

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

The FACILE-RS Python package allows to perform tasks around the archival and long term preservation of research software repositories. It consists in a set of Python scripts which facilitate the maintenance of software metadata by automating the generation of metadata in various formats from a unique metadata file that is maintained manually. FACILE-RS also makes it easier to publish and archive software releases according to the Open Science paradigm and the FAIR (Findable, Accessible, Interoperable, Reusable) principles for Research Software, by offering tools to automate the creation of releases and the upload to persistent research data repositories.

In particular, FACILE-RS allows to:

- create a DataCite record based on CodeMeta files present in repositories,
- create a CFF (Citation File Format) file from CodeMeta files,
- create archive packages in the BagIt or BagPack formats,
- create a release on the Develpment platform GitLab using the GitLab API,
- archive software releases using the RADAR service,
- use content from markdown files, bibtex files, or python docstrings to create web pages within the Grav CMS.

While the scripts can be run manually, they are designed to be used within GitLab CI/CD or another workflow automation system, in order to automate the process of maintaining metadata and creating persistent software releases.

Statement of need

Research software development is a fundamental aspect of academic research (Anzt et al., 2021), and it has now been acknowledged that the FAIR principles (Findable, Accessible, Interoperable, Reusable; (Wilkinson et al., 2016)), historically established to improve the reusability of research data, should also be applied to research software (Chue Hong et al., 2021). In particular, reproducible research requires that software and their associated metadata can be found easily by both machines and humans, and that they are retrievable via standardised protocols. In this context, several metadata standards are widely used across the scientific community:

- the Citation File Format (CFF) (Druskat et al., 2021) aims to indicate to users how to cite a software package,
- DataCite (DataCite Metadata Working Group, 2021) is a standard Metadata schema for archiving digital assets,
- CodeMeta (Jones et al., 2017) is an extension of schema.org created to standardize the exchange of software metadata across repositories and organizations.

The Journal of Open Source Software

All of these standards serve specific purposes and several of them are required to cover the whole software lifecycle. However, maintaining multiple metadata files in different formats represents a burden for research software developers, and can prevent them from adopting good software publication practices. In addition, as the content of the different metadata files is largely overlapping, maintaining these files manually can pose a risk to data consistency.

FACILE-RS aims to overcome these difficulties by making it easy to create and maintain the metadata associated to research software, as well as to publish software releases according to the FAIR principles on reputable research data repositories.

Functionality

The main prerequisite for using FACILE-RS in a software repository is to create a CodeMeta metadata file, for example using the CodeMeta generator. This metadata format is specifically tailored for describing scientific software.

The Python scripts that compose FACILE-RS are gathered in Table 1. While each of these scripts can be used individually and run manually, FACILE-RS was designed to be used within an automated workflow, for example using GitLab CI/CD, a tool for automating software development workflows via a continuous and iterative process.

Script	Functionality
`create_cff`	generates Citation File Format (CFF) metadata file
`prepare_release`	updates *version* and *dateModified* fields in metadata
`create_release`	creates release in GitLab
`create_datacite`	generates DataCite metadata file
`create_bag`	creates BagIt package
`create_bagpack`	adds DataCite XML to BagIt package
`prepare_radar`	reserves DOI on RADAR
`create_radar`	creates archive and uploads it to RADAR
`run_markdown_pipeline`	updates Grav CMS website
`run_bibtex_pipeline`	converts BibTex files and publishes references on Grav CMS website
`run_docstring_pipeline`	extracts docstrings from Python scripts and publishes them on Grav CMS website

Table 1: Components of FACILE-RS

A typical FACILE-RS workflow in GitLab CI/CD is illustrated on figure Figure 1. In this example, each time a commit is published, the different metadata files are automatically updated from the CodeMeta file.

This workflow also includes an automated process for creating software releases, both on GitLab and on the research repository RADAR. This process is triggered by creating a *pre-release* tag, which would be `pre-v0.1.0` for creating the release of version `v0.1.0`. During the *pre-release* phase, a DOI is reserved on RADAR and the software metadata associated with the release is updated. Once this is done, the proper release tag is automatically created, and the GitLab and RADAR releases are created.

Several tutorials for implementing such a workflow are provided within the FACILE-RS repository.

The Journal of Open Source Software

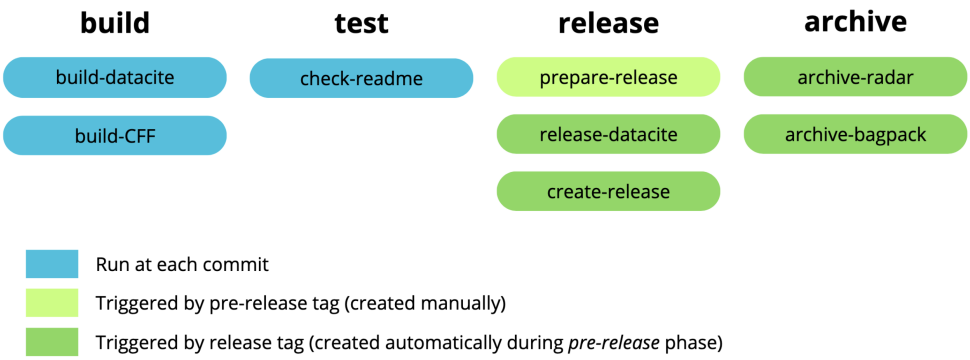

Figure 1: Typical structure of an automated FACILE-RS workflow

Acknowledgements

We acknowledge ...

References

Anzt, H., Bach, F., Druskat, S., Löffler, F., Loewe, A., Renard, B., Seemann, G., Struck, A., Achhammer, E., Aggarwal, P., Appel, F., Bader, M., Brusch, L., Busse, C., Chourdakis, G., Dabrowski, P., Ebert, P., Flemisch, B., Friedl, S., ... Weeber, R. (2021). An environment for sustainable research software in Germany and beyond: Current state, open challenges, and call for action. *F1000Research*, *9*(295). https://doi.org/10.12688/f1000research.23224.2

Chue Hong, N. P., Katz, D. S., Barker, M., Lamprecht, A.-L., Martinez, C., Psomopoulos, F. E., Harrow, J., Castro, L. J., Gruenpeter, M., Martinez, P. A., & Honeyman, T. (2021). *FAIR principles for research software (FAIR4RS principles)*. https://doi.org/10.15497/RDA00068

DataCite Metadata Working Group. (2021). *DataCite metadata schema documentation for the publication and citation of research data and other research outputs*. https://doi.org/doi.org/10.14454/3w3z-sa82

Druskat, S., Spaaks, J. H., Chue Hong, N., Haines, R., Baker, J., Bliven, S., Willighagen, E., Pérez-Suárez, D., & Konovalov, O. (2021). *Citation File Format* (Version 1.2.0). https://doi.org/10.5281/zenodo.5171937

Jones, M. B., Boettjiger, C., Mayes, A. C., Smith, A., Slaughter, P., Niemeyer, K., Gil, Y. G., Fenner, M., Nowak, K., Hahnel, M., Coy, L., Allen, A., Crosas, M., Sands, A., Hong, N. C., Cruse, P., Katz, D., & Goble, C. (2017). *CodeMeta: An exchange schema for software metadata. Version 2.0*. https://doi.org/10.5063/schema/codemeta-2.0

Wilkinson, M. D., Dumontier, M., Aalbersberg, Ij. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., Silva Santos, L. B. da, Bourne, P. E., & others. (2016). The FAIR guiding principles for scientific data management and stewardship. *Scientific Data*, *3*(1), 1–9. https://doi.org/10.1038/sdata.2016.18