Facilitating Reproducible In Silico Experiments with openCARP: a Step Toward FAIR and Open Science in Cardiac Electrophysiology

Marie Houillon[1], Jorge Sánchez[2], Matthias A F Gsell[3], Aurel Neic[4], Anton J Prassl[3], Gunnar Seemann[1], Gernot Plank[3,4], Edward J Vigmond[4,5], Axel Loewe[1]

[1] Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
[2] Universitat Politècnica de València, Valencia, Spain, [3] Gottfried Schatz Research Center, Division of Biophysics, Medical University of Graz, Graz, Austria
[4] NumeriCor GmbH, Graz, Austria, [5] Université de Bordeaux, Bordeaux, France

Abstract

While the philosophy of Open Science advocates accessibility and reusability of scientific outcomes, it can be challenging to apply these principles in computational studies, as research software is by nature modular and changing over time. This work aims to make it easy for researchers to create cardiac electrophysiology in silico experiments that can be shared and reproduced by other researchers, as well as published in persistent research data repositories.

For this purpose, we have implemented a feature in the openCARP cardiac electrophysiology simulation ecosystem, which allows users to create self-contained simulation bundles in a standardized structured format with related metadata. These bundles can then easily be shared and run on other platforms. A service for releasing experiments on the persistent research data repository RADAR is also provided.

Thanks to this bundle *feature, several openCARP simulation workflows were released in persistent research repositories already cited in subsequent publications.*

We believe that this tool will help promoting good practices in computational research by facilitating the publication of simulation workflows in accordance with the Open Science principles.

1. Introduction

The FAIR principles for research data [1] have been established in 2016 as a guide to apply the Open Science philosophy to modern publication practices and infrastructures. These guidelines aim to improve the *F*indability, *A*ccessibility, *I*nteroperability and *R*eusability of digital assets, both by humans and machines, as it is indispensable to rely on computational systems to deal with the increasing volume and complexity of data.

With the recognition of research software development as a fundamental aspect in research [2], it is now acknowledged that the FAIR principles, historically established for research data, should also be applied to research software. Yet, some particular characteristics of software as a digital asset, such as executability and evolution over time, have lead to propose dedicated FAIR for Research Software (FAIR4RS) principles in 2022 [3, 4]. They include the following recommendations:

- Software is assigned a globally unique and persistent identifier and different versions of the software are assigned distinct identifiers.
- Software is described with rich, searchable and indexable metadata, and both must be retrievable via standardised protocols.
- Software is both usable (can be executed) and reusable (can be understood, modified, built upon, or incorporated into other software).

With this work, we aim to make it easier for researchers to create and share cardiac electrophysiology simulations that comply with the FAIR4RS principles using the open cardiac electrophysiology simulator openCARP [5]: the *bundle* feature[1] that has been implemented within openCARP allows to create self-contained simulation bundles that can be easily shared and run by other researchers, as well as published in persistent research data repositories.

2. Methods

2.1. The openCARP ecosystem

The *bundle* feature has been implemented within openCARP, a cardiac electrophysiology simulation ecosystem with public source code offering multiscale simulations from ion channel to organ level [5]. openCARP is freely available for academic, non-commercial purposes.

[1] https://opencarp.org/community/upload-experiment

The full openCARP ecosystem contains several components allowing to design end-to-end cardiac electrophysiology in silico experiments including data pre- and post-processing as well as visualization, as illustrated in figure 1. Another important part of openCARP is its community platform, which fosters the communication between users and developers via a website used as a hub to access all resources around openCARP, a Question and Answer system, and a GitLab instance open to contributions.

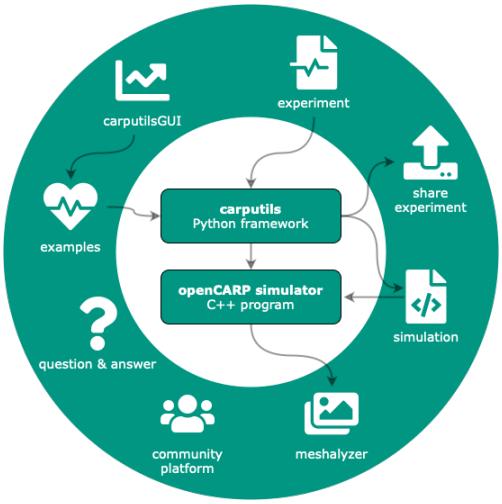

Figure 1. Components of the openCARP ecosystem. Interested readers are referred to [5] or www.openCARP.org for further details.

Within this environment, the Python-based framework carputils enables users to develop complex simulation pipelines, i.e. to automate in silico experiments including all modeling and simulation steps.

The *bundle* feature, which can be used for creating a self-contained bundle from an existing carputils experiment, was implemented as a submodule of carputils.

2.2. Functionality of the *bundle* feature

Starting from an existing carputils in silico experiment, the main functionality of the *bundle* feature is to turn the experiment into a self-contained simulation workflow that can be shared and run from another system which has openCARP installed.

In addition, when bundles are generated, they are provisioned with structured metadata information as well as a default license file, which make them suitable for publication in research data repositories. Concretely, we provide streamlined workflows to publish bundled carputils experiments in the research data repository RADAR[2].

[2]https://www.radar-service.eu

An overview of the whole workflow is illustrated in figure 2.

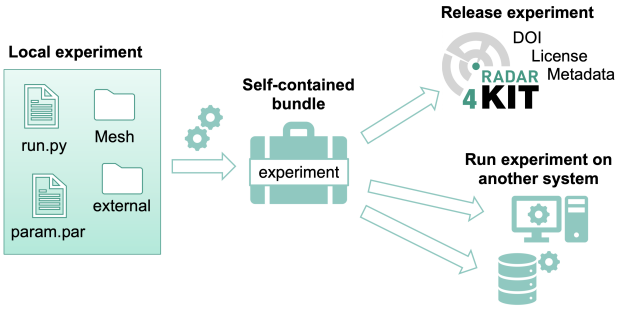

Figure 2. Overview of the *bundle* feature workflow.

2.2.1. Convert local experiment to self-contained bundle

carputils experiments that are created and run on a given system are in particular dependent on:

- the Python version
- the openCARP and carputils versions
- local files needed for executing the experiment
- command line options to customize the simulation

These constraints can make it difficult to ensure that a given experiment is reproducible by other users or on other systems.

Creating a self-contained bundle from an experiment minimizes the issues related to reusability and reproducibility. In particular, the bundle feature allows turning absolute paths that are used in the experiment's source code into relative paths within the bundle and to copy all files that are necessary to run the experiment into the bundle.

In addition, when the bundle is created, a metadata file following the DataCite Metadata Schema[3] is automatically generated and included in the bundle. This metadata file contains information facilitating findability and reusability of the simulation workflow:

- information about the bundle creator (name, email, ORCID ID)
- name of the experiment
- command line used to run the experiment when creating the bundle
- openCARP version used for running the experiment
- Python version used to create the bundle

The data asset resulting from this bundling workflow is a ZIP archive containing the self-contained experiment as well as the associated metadata.

[3]https://schema.datacite.org

2.2.2. Publish experiments according to the FAIR principles

The *bundle* feature does not only generate self-contained carputils experiments but also complies with the FAIR4RS principles for publication of research software. Thus, a bundle can be easily published in any research data repository. Within openCARP, we provide a service for publishing bundles directly in the research data repository RADAR.

The option provided in the *bundle* feature for releasing an experiment triggers to following workflow:

- The self-contained bundle of the in silico experiment is created.
- The bundle is populated with a metadata file following the DataCite Schema.
- The bundle is populated with a license file (Apache 2.0[4] by default).
- The content of the bundle is pushed to a Git repository of the user's choice.
- An email template is provided that can be sent to the openCARP curators.
- After validation of the bundle by the curators, the bundle will be published in the repository RADAR4KIT, using the metadata provided in the bundle, and the release will be associated with a persistent and citable DOI[5].

Uploading the bundle to a Git repository is recommended but optional: as an alternative, it is also possible to send a bundle in the ZIP format to the openCARP curators.

3. Using the *bundle* feature

Every carputils simulation workflow uses a Python script as an entry point, usually named `run.py`. A carputils experiment is then triggered by running this script, optionally with command line options for configuring the experiment.

Considering an experiment triggered with the command `./run.py`, a self-contained bundle for this experiment can be created using the following workflow:

- Local files used in the experiment's source code (configuration files, meshes, ...) are referred to using the `simfile_path()` function, for example:

```
cmd = carp_cmd(
     simfile_path('/path/to/simu.par'))
```

This ensures that when the bundle is created, the file `simu.par` is included in the bundle, and its path is adapted in the script included in the bundle.

- Add the `--bundle` command line option to the command used to run the simulation to create a bundle of the simulation:

```
./run.py --bundle my_first_experiment
```

This will create the self-contained bundle as `my_first_experiment.zip`.

To publish the experiment on RADAR4KIT, the following additional steps can be taken:

- Further customize metadata and license files.
- Create an empty public Git repository for the experiment with permission to push data to it.
- Trigger the release pipeline:

```
./run.py --bundle my_first_experiment
    --push-bundle <link-to-git-repo>
    --release-bundle
```

- Click on the generated link to send an email to the openCARP curators asking for a release of your experiment.

Using this workflow, the experiment will be pushed to the Git repository, which can be used by the openCARP curators to review your bundle before actually publishing it on RADAR.

4. Discussion

The *bundle* feature implements a standard data structure for carputils simulation workflows and proposes a publication workflow, which facilitates good practices outlined in the FAIR4RS principles. In particular, this solution addresses aspects of Findability and Accessibility:

- *Software is assigned a globally unique and persistent identifier*: the bundle is assigned a DOI when released on RADAR.
- *Software is described with rich metadata*: each bundle is automatically populated with a metadata file using the DataCite Metadata Schema, containing at least the minimal information required for publication in a research data repository according to the FAIR principles.
- *Metadata are searchable and indexable*: this is ensured by the use of the DataCite Schema for describing software metadata.
- *Software, and its metadata, is retrievable via standardised protocols*: this feature is provided by the RADAR service.

The *bundle* feature also facilitates strong compliance with the principle of Reusability. Indeed, during the bundle creation, all data necessary for running the simulation (such as source code, meshes or configuration files) is copied to the bundle, and the paths to these files are adapted consequently in the source code, so that the whole experiment is self-contained in the bundle. The reproducibility of the experiment is also strengthened by providing the command line options as well as the openCARP and Python versions used for running the simulation. As all release versions of openCARP are persistently archived and include cross-platform Docker images [6], this markedly facilitates reproduction.

[4] https://www.apache.org/licenses/LICENSE-2.0
[5] Digital Object Identifier, see https://www.doi.org

In addition, the *bundle* feature provides an automated way to convert existing carputils experiments to the structured format used in bundles, which makes it easy for creators of experiments to adopt this format and apply FAIR principles when sharing their simulation workflows.

Some researchers have adopted this approach already for releasing their in silico cardiac electrophysiology experiments: on RADAR4KIT, the data assets [7–9] have for instance been created and released using the *bundle* feature. An example of reference to a reproducible simulation workflow in a subsequent publication can be found in [10].

We believe that providing a structured format for cardiac electrophysiology simulation workflows as well as a convenient way to share this workflows can facilitate and incentivize exchange within the cardiac electrophysiology computing community and beyond. Future plans for functionality within openCARP include support for the versioning of experiments for better FAIR4RS compliance. The structured format that has been implemented could be exploited to offer even easier reproducibility of carputils experiments, for example using a Software as a service model, allowing to run bundled simulations via a web interface without the need to install openCARP locally to run carputils experiments. We hope that this work can serve as an example to inspire new initiatives promoting good practices for FAIR research software and Open Science in general.

Acknowledgments

This work was supported by the European High-Performance Computing Joint Undertaking EuroHPC under grant agreement No 955495 (MICROCARD) co-funded by the Horizon 2020 programme of the European Union (EU) and the German Federal Ministry of Education. This work was supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) under grant LO 2093/9-1.

References

[1] Wilkinson MD, Dumontier M, Aalbersberg IJ, et al. The fair guiding principles for scientific data management and stewardship. Scientific Data Mar 2016;3(1):160018. ISSN 2052-4463. URL https://doi.org/10.1038/sdata.2016.18.

[2] Anzt H, Bach F, Druskat S, et al. An environment for sustainable research software in germany and beyond: current state, open challenges, and call for action. F1000Research 2021;9(295). URL doi.org/10.12688/f1000research.23224.2.

[3] Chue Hong NP, Katz DS, Barker M, et al. FAIR principles for research software (FAIR4RS principles). Research Data Alliance 2022;URL https://doi.org/10.15497/RDA00068.

[4] Barker M, Chue Hong NP, Katz DS, et al. Introducing the fair principles for research software. Scientific Data Oct 2022;9(1):622. URL doi.org/10.1038/s41597-022-01710-x.

[5] Plank G, Loewe A, Neic A, et al. The openCARP simulation environment for cardiac electrophysiology. Computer Methods and Programs in Biomedicine 2021;208:106223.

[6] openCARP consortium, et al. openCARP v15.0, 2024. URL https://doi.org/10.35097/1979.

[7] Loewe A, Nagel C. Single human myocyte simulations of kcnq1 mutants (v1.0), 2022. URL https://doi.org/10.35097/580.

[8] Sanchez J, Amsaleg A, Loewe A. In silico characterization of the sinoatrial node (v2.0), 2022. URL https://doi.org/10.35097/763.

[9] Martinez Diaz P, Loewe A. The right atrium affects in silico arrhythmia vulnerability in both atria, 2023. URL https://doi.org/10.35097/1830.

[10] Amsaleg A, Sánchez J, Mikut R, Loewe A. Characterization of the pace-and-drive capacity of the human sinoatrial node: A 3d in silico study. Biophysical Journal 2022;121(22):4247–4259. URL https://doi.org/10.1016/j.bpj.2022.10.020.

Address for correspondence:

Institute of Biomedical Engineering
Karlsruhe Institute of Technology (KIT)
Fritz-Haber-Weg 1, 76131 Karlsruhe, Germany
publications@ibt.kit.edu