The Journal of Open Source Software

DARE Platform: a Developer-Friendly and Self-Optimising Workflows-as-a-Service Framework for e-Science on the Cloud

Iraklis A. Klampanos[*1], **Chrysoula Themeli**[1], **Alessandro Spinuso**[2], **André Gemünd**[3], **and Vangelis Karkaletsis**[1]

1 National Centre for Scientific Research "Demokritos" **2** Koninklijk Nederlands Meteorologisch Instituut **3** Fraunhofer-Institut für Algorithmen und Wissenschaftliches Rechnen (SCAI)

DOI: 10.21105/joss.02552

Software

- Review
- Repository
- Archive

Editor: Pending Editor

Submitted: 05 August 2020
Published: 05 August 2020

Introduction

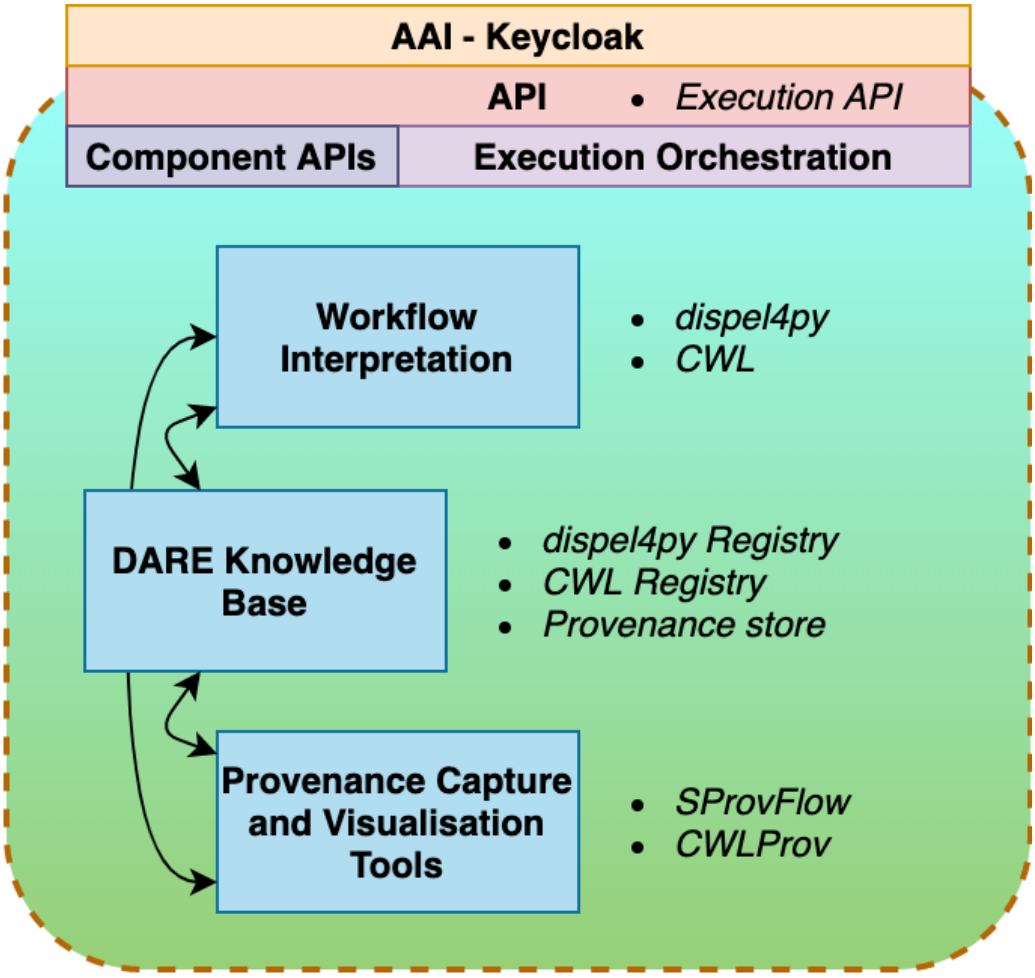
In recent years, modern science has relied more than ever on large-scale data as well as on distributed computing and human resources. Scientists and research engineers in fields such as climate science and computational seismology, constantly strive to make good use of remote and largely heterogeneous computing resources (HPC, Cloud, institutional or local resources, etc.), process, archive and analyse results stored in different locations and collaborate effectively with other scientists.

The DARE platform enables the seamless development and reusability of scientific workflows and applications, and the reproducibility of the experiments. More information on DARE can be found in (Atkinson et al., 2020, 2019; Klampanos et al., 2019).

The DARE Platform

The DARE platform is designed to live in-between user applications and the underlying computing resources. It is built on top of containerisation as well as parallelisation technologies, e.g. Kubernetes and MPI. Interfacing with client systems and end-users is achieved via RESTful APIs. The execution of scientific workflows is achieved via a Workflows-as-a-service layer, which can handle workflows described in either the dispel4py Python library(Filguiera, Krause, Atkinson, Klampanos, & Moreno, 2017), or in the Common Workflow Language (CWL)(Amstutz et al., 2016).

*Corresponding author.



The main components of the DARE platform are:

1. Workflow interpretation
    1. Dispel4py
    2. CWL
2. DARE knowledge base
    1. Dispel4py registry, which registers dispel4py workflows
    2. CWL registry, similar to the dispel4py registry for the CWL case
    3. Provenance store
3. Provenance capturing and visualisation
    1. s-ProvFlow, for acquisition and interactive exploration of data provenance
    2. CWLProv, model for capturing provenance information in CWL workflows, based on Khan et al. (2019).
4. DARE API
    1. Exposition of part of components' APIs, making use of Kubernetes Ingress.
    2. Execution API: a RESTful Web Service exposing dispel4py and CWL execution, while also providing basic file handling functionality.
    3. Keycloak API: Exposes AAI endpoints based on Keycloak, for modularity and ease of integration with external authentication mechanisms.

Developer-friendly WaaS

DARE offers user-friendly development, execution and monitoring of two complementary types of workflows: dispel4py and CWL. The platform accommodates the execution of such workflows in dynamic contexts, in the Cloud. Once a workflow has been developed and registered, it can be executed by name via RESTful endpoints.

Dynamic Loading of Execution Contexts

Execution environments are loaded on-demand, based on access of specific endpoints of the Execution API. Currently, the platform offers three different execution environments, for dispel4py workflows, CWL workflows and SPECFEM3D-Cartesian - a well-received code for simulating seismic wave propagation[1]. In this section, we introduce the CWL execution environment, as it is more widely recognisable, however the procedure is similar to the case of dispel4py.

Before users can make use of arbitrary, CWL-based execution environments they need to have the corresponding docker containers registered on the platform. The DARE platform installation administrator is responsible for testing and registering such environments, ensuring they are free from malicious software and that they behave as intended.

Once the execution environment, realised by a docker container, has been registered, scientists and research engineers can make use of it for their scientific workflows. The CWL registry allows users to register multiple bash and python scripts and associate them with specific docker containers. After registration, execution environments and CWL workflows are identified by name and version.

For executing a CWL workflow, The Execution API dynamically loads the corresponding execution environment and starts the workflow via the underlying Kubernetes container orchestration layer. DARE shields users from underlying implementation details whilst it allows them to share, restart or monitor their workflow-based applications.

Ease of use and monitoring

Alongside API-based access, DARE offers a testing environment, the *Playground*, where users can develop and test their workflows. Through the Playground API, we provide a simulation of the dispel4py workflow execution giving users immediate access to the logs and output files. In addition, users are provided with interactive tools to register and describe processing elements and complete workflows, which allows sharing, findability and reusability of methods. The platform also provides interactive provenance tools, enabling users to track workflow execution during run-time, and for the long-term management and validation of the experimental results Spinuso, Atkinson, & Magnoni (2019).

DARE Platform Use-cases

The DARE platform is currently used in the following domain applications:

1. Seismology: Rapid Assessment (RA) of ground motion parameters during large earthquakes.
2. Seismology: Moment Tensor 3D (MT3D) for ensemble-type of seismic modelling.
3. Volcanology: Ash fall hazard modelling.

[1] The SPECFEM3D endpoint is now obsolete, however we include it for completeness.

4. Climate-change: Extending Climate4Impact with efficient and transparent access to diverse computing resources.
5. Atmospheric sciences: Cyclone tracking and visualisation application.

Contributions of the DARE Platform

1. Interfacing with users and external systems via a comprehensive RESTful API
2. Facilitating the development of modular, reusable and shareable data-intensive solutions
3. Combining different workflow approaches, dispel4py and CWL, within the same platform and development environment
4. Via its execution API it orchestrates the dynamic spawning and closing of MPI clusters on the cloud for MPI-enabled components
5. It provides a flexible environment, which local administrators can parametrise, by supporting custom docker-based environments and user interfaces
6. It supports the collection, mining and visualisation of provenance information.

Software

The DARE platform is available on GitLab. We also have a GitLab page with installation instructions, API documentation and a short demo. A demo is available in the DARE Execution API GitLab Repository and can be used as an integration test.

Each DARE component typically include its own tests, client-side helper functions or a short jupyter notebook demo.

Future Work

Directions for future work include:

1. Make use of provenance data and workflow metadata to further automate the optimisation of workflow execution.
2. Provide wider-ranging search facilities to users for data, components and containerised environments, extending ongoing DARE work for searching over DCAT catalogues.
3. Provide of-the-shelf integration with domain-specific as well as generic repositories (e.g. with Zenodo) in order to facilitate better Open Science best practices.

Acknowledgements

This work has been supported by the EU H2020 research and innovation programme under grant agreement No 777413.

References

Amstutz, P., Crusoe, M. R., Tijanić, N., Chapman, B., Chilton, J., Heuer, M., Kartashov, A., et al. (2016). Common workflow language, v1. 0.

Atkinson, M., Filgueira, R., Gemünd, A., Karkaletsis, V., Klampanos, I., Koukourikos, A., Levray, A., et al. (2020, March). DARE architecture and technology internal report. Zenodo. doi:10.5281/zenodo.3697898

Atkinson, M., Filgueira, R., Klampanos, I., Koukourikos, A., Krause, A., Magnoni, F., Pagé, C., et al. (2019). Comprehensible control for researchers and developers facing data challenges. In *Proceedings of the 15th ieee international conference on eScience (to appear)*.

Filguiera, R., Krause, A., Atkinson, M., Klampanos, I., & Moreno, A. (2017). Dispel4py: A Python framework for data-intensive scientific computing. *International Journal of High Performance Computing Applications*. doi:10.1177/1094342016649766

Khan, F. Z., Soiland-Reyes, S., Sinnott, R. O., Lonie, A., Goble, C., & Crusoe, M. R. (2019). Sharing interoperable workflow provenance: A review of best practices and their practical application in cwlprov. *GigaScience*, *8*(11), giz095.

Klampanos, I., Davvetas, A., Gemünd, A., Atkinson, M., Koukourikos, A., Filgueira, R., Krause, A., et al. (2019). DARE: A reflective platform designed to enable agile data-driven research on the cloud. In *2019 15th international conference on eScience (eScience)* (pp. 578–585).

Spinuso, A., Atkinson, M., & Magnoni, F. (2019). Active provenance for data-intensive workflows: Engaging users and developers. In *2019 15th international conference on eScience (eScience)* (pp. 560–569). IEEE.