# CHAMP is a HPC Access and Metadata Portal

**Christopher Cave-Ayland**[1], **Michael Bearpark**[2], **Charles Romain**[2], **and Henry S. Rzepa**[*2]

**1** Imperial College London, Research Computing Service **2** Imperial College London, Department of Chemistry

**DOI:** 10.21105/joss.03824

**Software**

- Review
- Repository
- Archive

**Editor:** Chris Vernon

**Reviewers:**

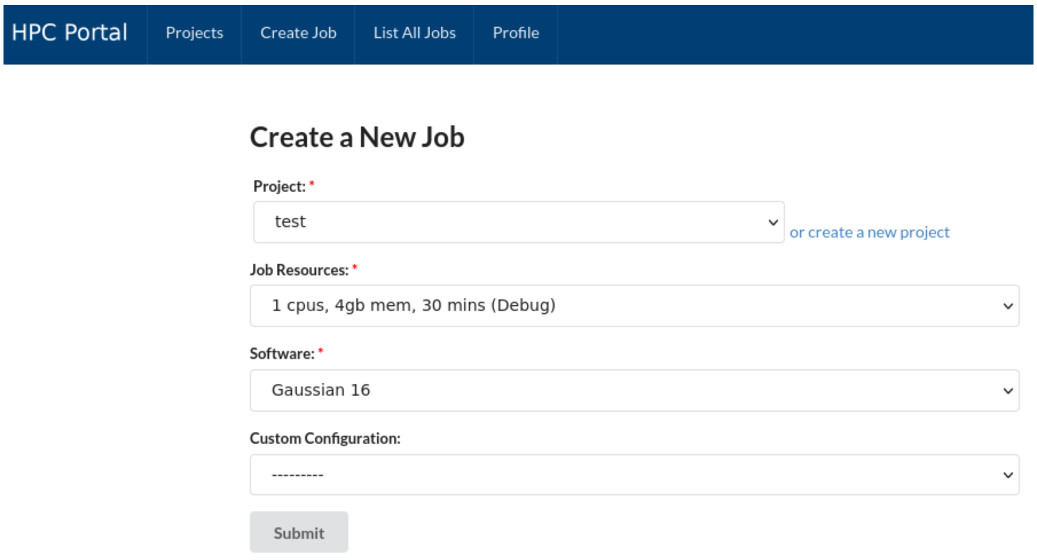
- @thurber
- @acrlakshman

**Submitted:** 11 October 2021
**Published:** 10 February 2022

## Summary

CHAMP is a high performance computing (HPC) and metadata portal which provides an easy to use workflow for FAIR (Findable, Accessible, Interoperable, and Reusabe) data generation and publication. It provides a web based interface allowing submission of HPC workloads and subsequent one-click publication of the results to data repositories such as Zenodo. Depositions support rich metadata to repositories that include a full implemention of the Subject property of the DataCite metadata schema (DataCite Metadata Working Group, 2021).

Users submit jobs simply by choosing from pre-configured software and computing resource specifications (see Figure 1).



Project: * test

Job Resources: * 1 cpus, 4gb mem, 30 mins (Debug)

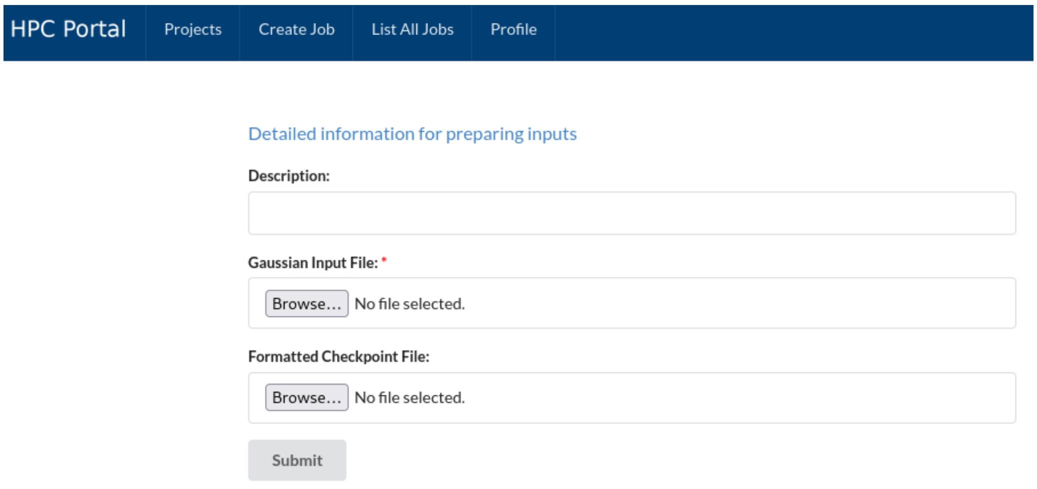
Software: * Gaussian 16

Custom Configuration: ---------

**Figure 1:** The form for the initial step in job creation

The user is then asked to upload the required input files for their chosen software and may optionally provide a job description (see Figure 2).

*corresponding author



Detailed information for preparing inputs

Description:

Gaussian Input File: *

Browse… No file selected.

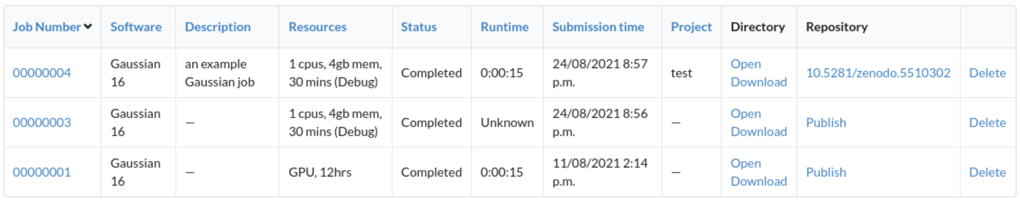
Formatted Checkpoint File:

Browse… No file selected.

Submit

**Figure 2:** The form for the file upload step in job creation

The status of jobs is available via a dedicated view (see Figure 3). Publication to data repositories is also possible once the job has completed, with the reserved persistent identifier (a DOI) shown associated with the job.



| Job Number | Software | Description | Resources | Status | Runtime | Submission time | Project | Directory | Repository | |
|---|---|---|---|---|---|---|---|---|---|---|
| 00000004 | Gaussian 16 | an example Gaussian job | 1 cpus, 4gb mem, 30 mins (Debug) | Completed | 0:00:15 | 24/08/2021 8:57 p.m. | test | Open Download | 10.5281/zenodo.5510302 | Delete |
| 00000003 | Gaussian 16 | – | 1 cpus, 4gb mem, 30 mins (Debug) | Completed | Unknown | 24/08/2021 8:56 p.m. | – | Open Download | Publish | Delete |
| 00000001 | Gaussian 16 | – | GPU, 12hrs | Completed | 0:00:15 | 11/08/2021 2:14 p.m. | – | Open Download | Publish | Delete |

**Figure 3:** Summary table of jobs run using the portal

CHAMP makes use of the Open OnDemand (Hudak et al., 2018) (OOD) framework. This allows CHAMP to be portable across different HPC systems and to integrate flexibly with institutional infrastructure (e.g., authentication mechanisms).

CHAMP represents over 15 years of experimentation in HPC portal design (Harvey et al., 2014). The recent 2.0 release is a ground up rewrite to facilitate its publication as an open-source package, to modernise the code base and to address isues of portability and sustainability.

## Statement of Need

Access to HPC resources has traditionally been available almost exclusively via command line interfaces which can present a barrier to entry for new and occasional users. The web interface provided by CHAMP is simple and intuitive providing an ideal entry point for non-experts.

Open OnDemand, the framework used by CHAMP, also provides a web based interface for HPC job submission. CHAMP differs by providing a higher level of abstraction that completely removes the need to deal with shell scripts or scheduler directives. This trade-off makes CHAMP much easier to use but introduces the restriction that only pre-configured software and resource configurations are supported.

The ability to publish job outputs to data repositories greatly simplifies the process of producing metadata-enabled FAIR datasets. The relevant files to upload and various rich metadata items are collected using workflows based on the selected software.

## Implementation and Features

CHAMP is written as a Passenger App within the OOD framework. It must therefore be deployed within a local OOD instance. Use of OOD provides numerous advantages such as portability, a strong security model and an active community of users. This allows the CHAMP code base to be quite minimal with mechanisms for authenticaton and system scheduler interaction provided by the parent framework. The per-user NGINX (Reese, 2008) architecture of OOD ensures all relevant computational processes are run under individual user identifiers.

CHAMP is written for use in Python (>3.6) using Django (Django Software Foundation, 2019) and supports extensive customisation via its main YAML (YAML Ain't Markup Language) configuration file. Administrators may configure the relevant software packages and resource configurations for their local system. There are no restrictions on the software or resource types that may be configured. Arbitrary workflows may be used to run software and to generate metadata. For example, this allows flexibility to use a job restart file if uploaded by a user or vary metadata for publication depending on job inputs.

The workflow that CHAMP imposes for HPC usage also provides inherent reproducibility. Individual jobs are structured within separate directories with relevant input files and scripts present. Jobs are recorded with metadata such as resources used and a free text description and are organised into projects. Job history can be filtered and searched according to this metadata allowing the portal to act as a simple electronic lab notebook.

An integration with Zenodo is included in the code base. Additional data repositories that support interaction via application programming interface and OAuth2 (Hardt, 2012) authentication may be configured via a plugin mechanism. This allows the development of custom integrations with institutional data repositories where desired, as in the case of Imperial College with an existing repository service (Harvey et al., 2017).

## Acknowledgments

We acknowledge the extensive contributions of Dr. Matthew Harvey for work on version 1 of this software.

## References

DataCite Metadata Working Group. (2021). *DataCite metadata schema for the publication and citation of research data and other research outputs v4.4*. https://doi.org/10.14454/FXWS-0523

Django Software Foundation. (2019). *Django* (Version 2.2) [Computer software]. https://djangoproject.com

Hardt, D. (2012). *The OAuth 2.0 Authorization Framework* (No. 6749). RFC 6749; RFC Editor. https://doi.org/10.17487/RFC6749

Harvey, M. J., Mason, N. J., & Rzepa, H. S. (2014). Digital data repositories in chemistry and their integration with journals and electronic notebooks. *Journal of Chemical Information and Modeling*, *54*(10), 2627–2635. https://doi.org/10.1021/ci500302p

Harvey, M. J., McLean, A., & Rzepa, H. S. (2017). A metadata-driven approach to data repository design. *Journal of Cheminformatics*, *9*(1). https://doi.org/10.1186/s13321-017-0190-6

Hudak, D., Johnson, D., Chalker, A., Nicklas, J., Franz, E., Dockendorf, T., & McMichael, B. (2018). Open OnDemand: A web-based client portal for HPC centers. *Journal of Open Source Software*, *3*(25), 622. https://doi.org/10.21105/joss.00622

Reese, W. (2008). Nginx: The high-performance web server and reverse proxy. *Linux J.*, *2008*(173).