Open OnDemand Overview and Customization

Alan Chalker, Ph.D.
Ohio Supercomputer Center
Columbus, OH
alanc@osc.edu

Julia Ma
Massachusetts Green HPC Center
Chicago, IL
jma@mghpcc.org

Jeff Ohrstrom
Ohio Supercomputer Center
Columbus, OH
johrstrom@osc.edu

Travis Ravert
Ohio Supercomputer Center
Columbus, OH
travert@osc.edu

Emily Moffat Sadeghi
Ohio Supercomputer Center
Columbus, OH
emoffat@osc.edu

Hazel Randquist
Ohio Supercomputer Center
Columbus, OH
hrandquist@osc.edu

Matt Walton
Ohio Supercomputer Center
Columbus, OH
mwalton@osc.edu

ABSTRACT

Open OnDemand Overview

Developed by the Ohio Supercomputer Center (OSC) and funded by the National Science Foundation, Open OnDemand (openondemand.org) is an open-source portal that enables web-based access to HPC services. Clients manage files and jobs, create and share apps, run GUI applications and connect via SSH, all from any device with a web browser.

Open OnDemand empowers students, researchers, and industry professionals with remote web access to supercomputers. From a client perspective, key features are that it requires zero installation (since it run entirely in a browser), is easy to use (via a simple interface), and is compatible with any device (even a mobile phone or tablet). From a system administrator perspective, key features are that it provides a low barrier to entry for users of all skill levels, is open source and has a large community behind it, and is configurable and flexible for user's unique needs.

Relevance to the Gateways Community

OOD is currently installed and running at 1,600+ academic, governmental, and commercial HPC centers both in the US and internationally, including the University of Utah, Stanford, Pittsburgh Supercomputer Center, Tufts University, University at Buffalo, University of Arizona, Texas A&M, and NVIDIA, and is actively being evaluated by many more. Most of these centers send representatives to the Gateways conferences.

The OOD team has held tutorials at multiple PEARC and Gateways conferences in recent years as well as regular online webinars, each of which have seen significant attendance of many dozens of people. As the Gateways series of conferences has historically had many attendees from locations that utilize OOD, as well as attendees from many of the most prolific science gateways development teams, holding another tutorial is a natural fit for the conference.

This proposed tutorial is meant to follow that same general format as utilized at PEARC and Gateways in the past with regards to application development and integration, with some key updates to reflect the changes in the Open OnDemand ecosystem over the past year.

A key differentiator of this year's BOF compared to previous years is that the Open OnDemand project has new funding from the NSF POSE program, which is designed to establish a new governance model for the ecosystem. Open OnDemand is in the process of establishing a steering committee and working groups, which is of significant interest to a large portion of the Gateways24 attendees.

An additional notable differentiator from previous years is the release in early 2024 of version 3.1 of the platform, that has many additional features that are of interest to the community, including integration with Globus and a new project manager.

Details of each of these key features will be included in the presentation.

Tutorial Overview:

This tutorial is meant to provide attendees with the basic tools to allow them to develop or integrate new applications into Open OnDemand. Nearly any software application can be made accessible via OOD. The official OOD github repo currently has links to software that appeals to a wide range of scientific disciplines, such as Jupyter, Abaqus, ANSYS, COMSOL, MATLAB, RStudio, Tensorboard, QGIS, VMD, RELION, STATA and Visual Studio. The OOD development team is also aware of planned or ongoing work to integrate many other software packages and platforms, including many that are prominent within the Science Gateways community, such as Galaxy, TAPIS, Globus, and Pegasus.

A general agenda is as follows:

Introduction to Open OnDemand:

~5 minutes (lecture & demonstration)

A very basic introduction to OnDemand will be presented to provide attendees with the goals of using OnDemand later in the session.

Open OnDemand Hands-on Introduction:

~15 minutes (hands-on activity)

The hands on walk through will provide details of how to use OnDemand and serve as an example of user training. We will walk users through several steps including how to log in as well as using the Files, Job Composer, Active Jobs, Shell, and the File Editor App. We will walk users through a similar tutorial showing how to launch multiple interactive Apps. Participants will gain a working knowledge of how to use Open OnDemand.

Open OnDemand Customization:

~45 minutes (hands-on activity)

The instructor will walk attendees through basic configuration of the Open OnDemand dashboard. This will go through basic configurations like navigation bar color to more complex additions like adding html to the page. The instructor will then review how apps work with some detail to familiarize the participant with the concepts in this tutorial. The hands-on tutorial will then guide the participant through getting a Jupyter app to work in the environment. Participants will then extend and customize this app through detailed instructions.

Open Floor Discussion:

~20 minutes

The remainder of the time will be dedicated to open floor discussions and questions from the participants regarding specific ideas or issues they may have.

The tutorial will end with a short PowerPoint presentation that provides attendees with options for contacting the instructors and other software developers at OSC. We will also discuss future development plans, invite attendees to participate in the community, and answer questions.

Tutorial Logistics:

This is a 90 minute long, hands-on, step-by-step instructional tutorial utilizing a mixture of PowerPoint presentations, brief pre-recorded videos, and demonstrations. Attendees will utilize pre-configured Docker containers and materials published on GitHub to step through key aspects of the semi-automated installation/configuration process on their own machines. Because all the materials are publicly available, they can repeat the process outside the tutorial at their leisure and share it with colleagues.

Attendee Skill Level:

Intermediate

Tech Requirements:

Attendees must utilize their own laptop with the Docker client preinstalled and tutorial containers pre-downloaded. Docker experience is helpful, but not required. Linux command line experience and an understanding of HPC clusters and batch scheduling is required.

Keywords—Open OnDemand, App Development, HPC

REFERENCES

[1] Wisniewski, L, Chalker, A, et al. (2023) Augmenting the User Experience in Open OnDemand. PEARC '23: Practice and Experience in Advanced Research Computing. https://doi.org/10.1145/3569951.3597546

[2] Chalker, A, Settlage, R, Hudak, D. (2021) Open OnDemand App Development and Integration. Gateways 2021. https://doi.org/10.5281/zenodo.5570225

[3] Settlage, R, Chalker, A, et al. (2021) Open OnDemand as a Platform for Virtual Learning in Higher Education. Proceedings of Sixth International Congress on Information and Communication Technology. https://doi.org/10.1007/978-981-16-1781-2

[4] Chalker, A, et al. (2020) Open OnDemand: State of the platform, project, and the future. Concurrency and Computation: Practice and Experience. https://doi.org/10.1002/cpe.6114

[5] Settlage, R, Chalker, A, et al. (2020) Portals for Interactive Steering of HPC Workflows. Tools and Techniques for High Performance Computing. https://doi.org/10.1007/978-3-030-44728-1_11

[6] Settlage, R, Chalker, A, et al. (2019) Open OnDemand: HPC for Everyone. International Conference on High Performance Computing. https://doi.org/10.1007/978-3-030-34356-9_38

[7] Franz, E, Chalker, A, et al. (2019) Scaling R Shiny Apps to Multiple Concurrent Users in a Secured HPC Environment Using Open OnDemand. Proceedings of the Practice and Experience in Advanced

Research Computing on Rise of the Machines. https://doi.org/10.1145/3332186.3332211

[8] Rodgers, M, Chalker, A, et al. (2019) Data Commons to Support University-Wide Cross Discipline Research. Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines. https://doi.org/10.1145/3332186.3335198

[9] Nicklas, J, Chalker, A, et al. (2018) Supporting distributed, interactive Jupyter and RStudio in a scheduled HPC environment with Spark using Open OnDemand. Proceedings of the Practice and Experience on Advanced Research Computing. https://doi.org/10.1145/3219104.3219149

[10] Hudak, D, Chalker, A, et al., (2018) Open OnDemand: A web-based client portal for HPC centers. Journal of Open Source Software. https://doi.org/10.21105/joss.00622

How to Spin Up a Short-term Cloud Science Platform for Workshops

Alex Antunes
Johns Hopkins Applied Physics Laboratory
Laurel, USA
0000-0002-3098-2602

Peter Shumate
Johns Hopkins Applied Physics Laboratory
Laurel, USA
peter.shumate@jhuapl.edu

Shawn Polson
Laboratory for Atmospheric and Space Physics
University of Colorado Boulder
Boulder, USA
shawn.polson@lasp.colorado.edu

***Abstract*— We discuss approaches for instantiating a short-term multi-user cloud for a large audience of cloud newcomers to teach better research software engineering skills within a domain-specific environment. We walk through our PyHC Summer School environment using the HelioCloud open source platform as the example, and discuss the logistics required and lessons learned.**

Keywords— gateways, cloud, research software engineer, containers, heliophysics

I. INTRODUCTION (*HEADING 1*)

We discuss gateway management approaches for transitory cloud environments including logistics, costs, guardrails, difficulties in syncing containers with outside tutorial contributors, risks, and benefits. We have spun up a week-long cloud environment to train early careers in the specific software and problems for heliophysics research, then torn it down afterwards. Each time we run this "PyHC Summer School", we improve our 'one button deploy' (spoiler: more than one button involved) and get better at container management to support the dozen sometimes conflicting science packages required by the environment. Success requires a mix of good technical basis and really structuring how both contributors and participants will interact with the system.

Our goal for running PyHC workshops is that engaging in research as a scientist or research software engineer requires domain knowledge, programming skills, and access to big datasets and enough compute power to grind through them. Building these skills gets boosted when experienced experts guide hands-on sessions within a uniform environment. Hence an open source cloud + container + data sources customized for the domain, with an easy interface such as Jupyter Notebooks, and sufficient cloud performance to scale.

However, running such an event efficiently involves crafting infrastructure, gathering tutorials from disparate community users, and creating a secure yet easily accessible environment that need only be ephemeral for the duration of the event. Unlike onboarding long-term users or creating tools for existing working groups, the logistics of spinning up a temporary yet fully functional scientific platform with domain-required software pre-loaded and tested are complex.

Specifically because such events are short-term and ephemeral, the risk profile is different than an institutional research environment. Concerns include the need for a pop-up architecture requiring minimal dev time to deploy, difficulties in creating a stable container that supports multiple presenter needs with potential package incompabilities, and dealing with costs and guardrails for the runtime. The design should minimize hurdles in attendees mastering the interface while limiting cost risk and exposure for the provider.

Our HelioCloud [1] platform is a Pangeo-derived open source set of cloud software for the heliophysics research community, available on github with details at heliocloud.org. Collaborating with the 'Python in Heliophysics Community' (PyHC.org, [2]) community, we've supported spinning up a Jupyter Daskhub AWS-backed framework for their week-long summer schools, where 400+ participants learn how to use both 'the cloud' and the scientific Python packages needed for research software engineering via hands-on problems presented by package developers that draw on real science problems. It is aimed at graduate students, early career scientists, and anyone eager to deepen their understanding of Python in the Heliophysics and Space Weather disciplines.

In our 90 minute tutorial, we walk through the basics needed to:

(a) install a multiple user AWS cloud environment with easy sign on and Jupyter notebooks, backed by a single AWS account

(b) resolve conflicting dependencies for the supporting container build (an iterative process, alas)

(c) quickly onboard the participants with accounts pre-loaded with the content tutorials so everyone can try it from the user perspective

(d) discuss the runtime support needed for the inevitable individual problems

(e) discuss the cost risks we're willing to accept and the steps we took to mitigate these

(f) provide statistics on usage, costs, and outcomes

We also invite questions and sharing of other solutions people have used.

1) Proposed length: 90 minutes

2) Recommend skill level:

Beginner to 'cloud', intermediate familiarity with containers and/or Python environments

3) Technology and/or Software requirements: Web connection required. We will have a pop-up environment set up with demo Notebooks (some including deliberate bugs) to provide an example runtime experience, and provide examples of the reporting cost dashboards the administrator will use.

References

[1] "HelioCloud: a community cloud-based approach to heliophysics analytics and software development. Thomas, B.A. et al, 2022, 2022AGUFMSH45B..01

[2] Python in Heliophysics Community (PyHC) Standards, 10.5281/zenodo.2529130

Cheap and FAIR: Building a Serverless Research Data Repository

1st Andrea Zonca
San Diego Supercomputer Center
University of California San Diego
La Jolla, CA USA
0000-0001-6841-1058

2nd Rick Wagner
San Diego Supercomputer Center
University of California San Diego
La Jolla, CA USA
0000-0003-1291-5876

***Abstract*—In this tutorial we will walk our attendees through the process of starting from data files in a folder, organizing them into a Globus collection and then publish a data catalog website to GitHub pages using the "Serverless Research Data Repository" (SRDR) tools.**

At each stage of the process we will highlight the features required to make a published dataset compliant to the FAIR (Findability, Accessibility, Interoperability, and Reuse) principles.

The tutorial is targeted both at gateway developers interested in deploying and administering their own data portal and at attendees interested in publishing their data in the most effective way.

***Index Terms*—Science Gateways, FAIR, Globus, GitHub, Markdown**

I. Tutorial Specifications

TABLE I
TUTORIAL LENGTH, LEVEL, AND REQUIREMENTS

Tutorial Length	3 hours
Recommended Skill Level	Intermediate
Prerequisites	Familiarity with GitHub and Markdown
Technical Requirements	A web browser and either SSH or a terminal with `git` and Python. GitHub and Globus accounts are needed (both are free).

II. Serverless Research Data Repository Model

The Serverless Research Data Repository (SRDR) architecture was created to meet four goals common to the CMB-S4 Collaboration[1] and other research projects:

- establishing data repositories which incorporate best practices;
- reducing or eliminating operational costs for projects to maintain a data repository;
- providing new projects with a base of good data practices;
- streamlining the later publication or curation of datasets.

While there are several related terms used to describe a data repository, including archive, data portal, and registry, the SRDR is based on this definition:

> A data repository is a set of *datasets* organized in and accessible from a *collection*, described by and discoverable in a *catalog*, and managed by one or more *policies*.

These four components provide a conceptual framework for research projects to organize their data practices. We also make the premise that the *quality* of a data repository is based on the policies it adopts (e.g., metadata standards and availability) and how well it implements them.

III. SRDR Architecture

The SRDR architecture, shown in Figure II, is based on the goals, definition, and premise delineated in the previous section. The SRDR combines Globus Guest Collections [1] for data storage and access (the collection) and GitHub [2] Pages for data discovery and description (the catalog) with policy recommendations as a data repository template requiring minimal effort to adopt and operate. Like the Django Globus Portal Framework [3] the SRDR architecture is derived from the Modern Research Data Portal [4] design pattern which provides a robust system of decoupled components. Many projects can deploy a data repository based on the SRDR without incurring any ongoing operational costs. An inspiration for this design is X-ray Tomography Data Bank, or TomoBank [5], which uses Read the Docs to publish a catalog of datasets originally hosted on Petrel at the Argonne Leadership Computing Facility [6].

The SRDR model has been deployed by the authors to build the Data Portal for the CMB-S4 experiment in the field of Cosmology to share maps of the Universe in the microwave regime both within the scientific collaboration and publicly: data.cmb-s4.org. Fig. VII-A shows a screenshot of the CMB-S4 data portal homepage and a page containing list of files identified by their own metadata and containing links to download the actual files via HTTPS through Globus.

IV. Tutorial Goals

The two main goals of the tutorials are to teach the attendees the most important aspects of the FAIR principles and apply them in practice by deploying their own data portal using the SRDR tools.

[1] https://cmb-s4.org

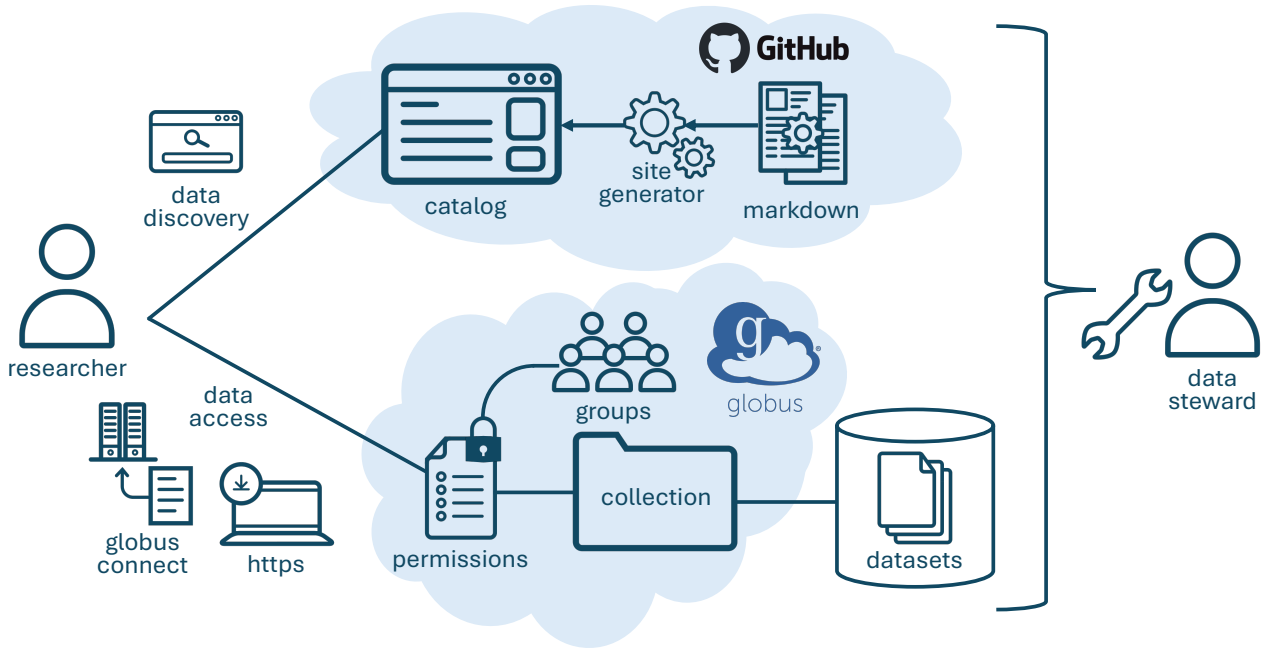


Fig. 1. Overview of the SRDR architecture.

The tutorial will cover many general aspects of data repositories, also from the users' perspective. Therefore, it is not only targeted to gateway developers but also to scientists who want to learn how to publish their own datasets in the most effective way.

After this tutorials, attendees will:

- understand the basic components of a research data repository;
- understand the FAIR data guidelines;
- know the difference between a dataset's files, metadata, identifier, and landing page;
- the role of machine-readable metadata in data discovery;
- be able to apply access control policies to datasets and their metadata;
- know how to use the tools provided by SRDR.

V. Prerequisites

Prior to the tutorial, attendees should have a GitHub (github.com) and Globus (globus.org) accounts. Attendees can reuse their accounts. When creating a Globus account, users can select GitHub to login, so creating a GitHub account first is recommended.

Attendees will have the option to download the sample datasets and run the required scripts on their own laptops. However, it is recommended to connect via SSH to the AWS EC2 instances provided during the tutorial. This will ensure a consistent environment for all attendees.

We will provide pre-configured Globus Guest Collections and GitHub repositories with GitHub Pages enabled.

VI. Tutorial Agenda

TABLE II
Tutorial Agenda

Time	Topic
0:00 - 0:15	Introduction & Setup
0:15 - 0:30	Background: The SRDR Model & Components
0:30 - 0:45	Prepare Datasets
0:45 - 1:00	Background: FAIR Data
1:00 - 1:30	Create the Collection
1:30 - 2:00	Break
2:00 - 2:15	Background: Identifiers & Landing Pages
2:15 - 2:45	Create the Catalog
2:45 - 3:00	Enhancing the Catalog
3:00 - 3:15	Demos: Extending the SRDR
3:15 - 3:30	Discussion & Wrapup

VII. Tutorial Content

A. Introduction on SRDR

In the first part of the tutorial we will present the SRDR architecture, as shown in Figure II, we will focus on the role played by Globus (data hosting and permissions) and by GitHub (website hosting and metadata catalog).

We will highlight how the SRDR model is designed to be cheap both in terms of operational costs and in terms of maintenance effort. This is achieved by relying on institutional data storage and Globus services, generally free for academic users, and on GitHub Pages for hosting the static catalog website.

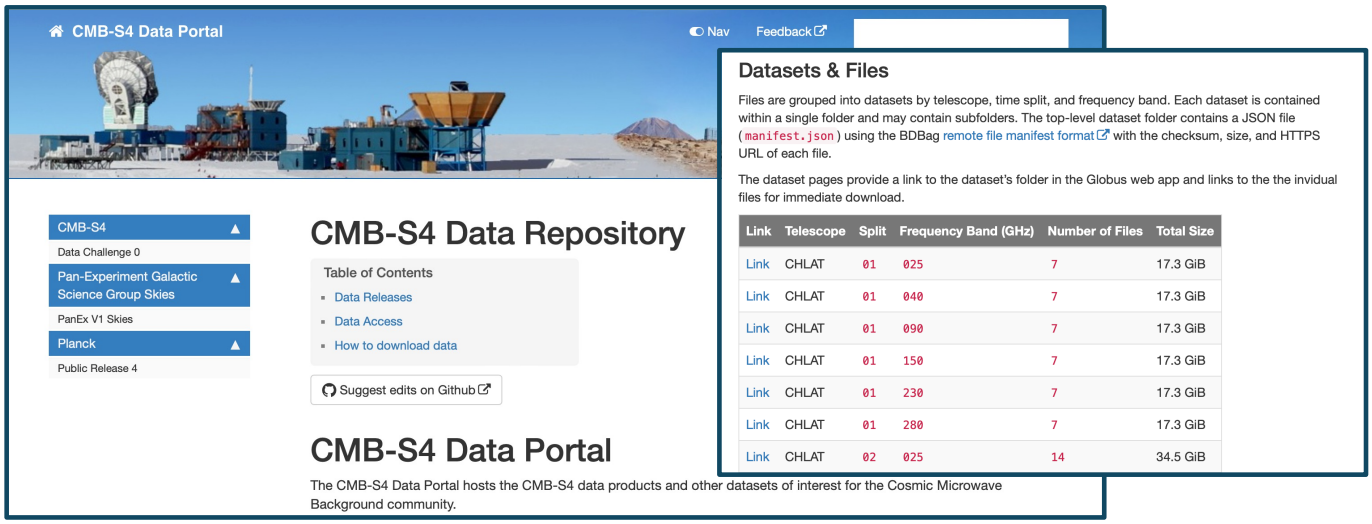


Fig. 2. Screenshot of the CMB-S4 Data Portal, the *catalog* component of SRDR, and a list of datasets.

B. The Datasets and Collection

We will provide example datasets with an open license that the attendees will work with. They will start by packaging the datasets adhering to FAIR principles, including defining an identifier and generating metadata for each dataset. The attendees will ingest the datasets into a Globus Guest Collection that will be shareable through the Globus platform. The attendees will also learn how to configure the Globus Guest Collection with permissions suitable to their needs and to expose the data for HTTPS access. Finally, they will run a tool to extract minimal metadata about the files that will be shared both via Globus and via the Data Portal to enable automated data retrieval.

C. The Catalog

Next the attendees will learn how to use SRDR tools to automatically create Markdown files to build the data catalog as a statically generated website. We will introduce concepts of FAIR related to the catalog and establish nomenclature of its main components. The attendees will then push their Markdown files to GitHub to be automatically compiled by a GitHub Action into a website using Jekyll [7] and published via GitHub Pages. They will be able to navigate through the data catalog on their browser and verify they can download data files directly to their laptop with no need of having Globus installed. Finally we will go though a quick demo on how to build more advanced features on top of SRDR, for example doing visualization and basic analysis right in their browsers.

VIII. Instructors

Andrea Zonca is the lead of the Scientific Computing Applications Group at the San Diego Supercomputer Center (SDSC), he has a background in Cosmology and is a member of CMB-S4. He is in charge of maintaining the Data Portal for the CMB-S4 collaboration.

Rick Wagner is the Chief Technology Officer (CTO) at SDSC. Previously, Rick was the Globus Professional Services Manager where he led the development of several data portals and architected the Django Globus Portal Framework. He has supported FAIR data within astrophysics, climate science, and bioinformatics.

Acknowledgment

This research used resources of the National Energy Research Scientific Computing Center (NERSC), a Department of Energy Office of Science User Facility using NERSC award HEP-ERCAP0023125. Support was also provided by UC San Diego Research IT Services [8].

References

[1] K. Chard, I. Foster, and S. Tuecke, "Globus: Research data management as service and platform," in *Proceedings of the Practice and Experience in Advanced Research Computing 2017 on Sustainability, Success and Impact*, ser. PEARC '17. New York, NY, USA: Association for Computing Machinery, 2017. [Online]. Available: https://doi.org/10.1145/3093338.3093367

[2] GitHub, "Github pages — websites for you and your projects." https://pages.github.com, 2024, [Online; accessed 9-April-2024].

[3] N. Saint, R. Chard, R. Vescovi, J. Pruyne, B. Blaiszik, R. Ananthakrishnan, M. Papka, R. Wagner, K. Chard, and I. Foster, "Active research data management with the django globus portal framework," in *Practice and Experience in Advanced Research Computing*, ser. PEARC '23. ACM, Jul. 2023. [Online]. Available: http://dx.doi.org/10.1145/3569951.3593597

[4] K. Chard, E. Dart, I. Foster, D. Shifflett, S. Tuecke, and J. Williams, "The modern research data portal: a design pattern for networked, data-intensive science," *PeerJ Computer Science*, vol. 4, p. e144, Jan. 2018. [Online]. Available: http://dx.doi.org/10.7717/peerj-cs.144

[5] F. D. Carlo, D. Gürsoy, D. J. Ching, K. J. Batenburg, W. Ludwig, L. Mancini, F. Marone, R. Mokso, D. M. Pelt, J. Sijbers, and M. Rivers, "Tomobank: a tomographic data repository for computational x-ray science," *Measurement Science and Technology*, vol. 29, no. 3, p. 034004, feb 2018.

[6] W. E. Allcock, B. S. Allen, R. Ananthakrishnan, B. Blaiszik, K. Chard, R. Chard, I. Foster, L. Lacinski, M. E. Papka, and R. Wagner, "Petrel: A programmatically accessible research data service," in *Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (Learning)*, ser. PEARC '19. New York, NY, USA: Association for Computing Machinery, 2019.
[7] Jekyll Core Team, "Jekyll," 2024. [Online]. Available: https://jekyllrb.com/
[8] UCSD Research IT, "Research it services," https://research-it.ucsd.edu, 2024, [Online; accessed 14-April-2024].

Creating Science Gateway or HPC portal on OneSciencePlace

Duration: 90 mins
Recommended skill: Beginner
Requirements: Web browser
Project website: OneSciencePlace.org

OneSciencePlace is a new platform to build an online and composable cyberinfrastructure that aims to transform delivery of FAIR content and computing in a single and easy to use environment. The platform could be used to build Science Gateways, HPC portals, Data repositories, Knowledgebase, and other highly customized applications. OneSciencePlace platform is built on a set of mature technologies that includes Drupal, SeedMeLab, Tapis and others. It can seamlessly interface with more than one compute resources such as Linux hosts, HPC clusters as well as data resources such as POSIX file systems or S3 object storage. Integration with other options such as Kubernetes and Globus are also on the future development roadmap.

This tutorial will focus on Science Gateway and HPC portal use cases and will guide attendees to explore OneSciencePlace in a hands-on manner. Each of the following aspects will be discussed/demonstrated in the tutorial with hands on component:

- Overview of available compute and data systems
- Using predefined applications (apps) (Hands-on)
  - Containerized web app such as Jupyter/RStudio
  - Containerized VNC app such as Linux Desktop
  - Command line executable on a system
- Use above apps on a Linux host or an HPC cluster (Hands-on)
- Creating and using a apps along with their user interface (Hands-on)
- Adding a new compute system
- Authentication options including local accounts, LDAP or Single-sign-on
- FAIR publishing and data sharing

Attendee outcomes
Attendees will be able to gain hands-on experience of OneSciencePlace. They will learn to use apps and data management on more than one system, in a single environment. Attendees will gain an understanding of creating new apps along with its user interface with no-code. The tutorial will help germinate ideas on how OneSciencePlace could be used to serve as a single environment that seamlessly integrates to more than one cluster/host and provides users an ability to run a variety of applications. They will also gain an understanding and feasibility allowing users to bring and add their own applications on OneSciencePlace that is scalable and empowering to users.

Reproducible ML Workflows and Deployments with Tapis

Joe Stubbs[1] , Nathan Freeman[1], Anagha Jamthe[1], Christian Garcia[1] , Dhanny Indrakusuma[1]

Texas Advanced Computing Center, Austin, TX, USA
{jstubbs, nfreeman, ajamthe, cgarcia}@tacc.utexas.edu, dhannywi@utexas.edu

Abstract:

This tutorial aims to provide researchers with a comprehensive introduction to the latest reproducible Machine Learning (ML) workflows and tooling provided by the NSF-funded Tapis v3 Application Programming Interface (API) and User Interface (UI). Through hands-on exercises, participants will gain experience in querying pre-trained ML models from public ML infrastructures such as Hugging Face or the ICICLE AI Institute, deploying various ML work-environments, and developing ML workflows on national-scale supercomputing resources. Throughout this tutorial, we will focus on utilizing the Tapis Workflows API, Tapis Pods API, Tapis ML Hub API, and TapisUI. These production-grade services are designed to simplify the facilitation and creation of trustworthy, reproducible, scientific workflows. By abstracting the complexities of underlying technologies behind user-friendly APIs, the Tapis services enable seamless integration with high performance computing (HPC) resources available at institutions with a Tapis deployment. Ultimately the tutorial aims to empower researchers to efficiently develop, deploy, and maintain their own ML workflows.

Description and Format:

We will set up training accounts for attendees so they can access Tapis resources regardless of their current TACC account creation status. Course material, including the slides and hands-on exercises for the tutorial, will be published on Github pages and will remain available to the attendees during and after the tutorial.
The section *Introduction to Tapis* will include theoretical concepts followed by simple hands-on-exercises. Attendees will be able to run the exercises in both Jupyter Notebooks and TapisUI. TapisUI is a serverless interactive science gateway that runs entirely on GitHub pages to interact with different Tapis services. A pre-computed example Jupyter notebook containing a description of building and running the scientific workflows will be available in case of internet outage. The tutorial will have a good mix of presentation and hands-on exercises for the attendees to learn and implement their own scientific computing workflows. We will have proctors working throughout the session to help attendees with the tutorial. The proposed tutorial schedule is as follows:

Time	Block	Topics
40 min	1	Introduction to Tapis, Authentication, TapisUI Overview
20 min	1	Introduction to Tapis Pods and deploy Pod instances
30 min	1	Introduction to Tapis ML Hub and interfacing with models
30 min	-	Break
25 min	2	Introduction to Tapis Workflows
20 min	2	Creating an ML workflow to facilitate automatic data collection
25 min	2	Overview of more Tapis use cases and integrations
10 min	2	Potential Tapis Use Cases (discussion with attendees)
10 min	2	Q/A

Tutorial Duration: 3 hours (excluding break)

Learning Outcomes:

By the end of this workshop attendees will be able to:

- Use Tapis APIs for creating and executing ML workflows
- Interact with Tapis ML Hub and interface with models
- Deploy instances with the Pods API which utilize GPU resources
- Launch ML oriented JupyterHub instances alongside other ML containers
- Understand how to create defined workflows for a real-world scientific use-case
- Have a basic understanding of deploying LLM models from Hugging Face
- Utilize Tapis UI to interact with Tapis services

Target Audience:

The audience for this workshop fits into three categories:

1) Researchers that utilize national, campus and local cyberinfrastructure resources and wish to do so in a reproducible, scalable and programmable manner.

2) Cyberinfrastructure specialists such as research software engineers (RSE), gateway providers/developers and infrastructure administrators. People in these roles can utilize open source technologies and state-of-the-art techniques to enable portable, reproducible computation.

3) Cyberinfrastructure directors, managers and facilitators that are looking for solutions to aid and educate their institutional researchers in order to better leverage local and distributed computational and cyberinfrastructure resources.

Content Level:

Beginner 70%, Intermediate 30%

Audience Prerequisites:

Attendees should have DockerHub accounts. Attendees must use their own laptop for the hands-on part of the tutorial. Attendees should have TACC accounts or use day-of training credentials.

Acknowledgement:

This workshop material is based upon work supported by the National Science Foundation Plant Cyberinfrastructure Program (DBI-0735191), the National Science Foundation Plant Genome Research Program (IOS-1237931 and IOS-1237931), the National Science Foundation Division of Biological Infrastructure (DBI-1262414), the National Science Foundation Division of Advanced CyberInfrastructure (1127210), (2112606), and (1931439 and 1931575).

Presentation team:

Dr. Joe Stubbs is a Research Associate and leads the Cloud and Interactive Computing (CIC) group at the Texas Advanced Computing Center at the University of Texas at Austin. Dr. Stubbs is currently the PI of two NSF-funded projects and has played a fundamental role in developing numerous national-scale cyberinfrastructure systems for various scientific and engineering communities used by thousands of researchers.
Previous trainings:

- PEARC 23: Best Practices of CI/CD for High Performance Computation with Tapis Workflows API
- PEARC 22: Building Portable, Scalable and Reproducible Scientific Workloads across Cloud and HPC for Gateways
- TACCster 2022: Tapis day at TACC
- Gateways 21: Portable, Scalable, and Reproducible Scientific Computing: from Cloud to HPC
- PEARC 2020: Leveraging Tapis For Portable, Reproducible High Performance Computing In the Cloud
- Gateways 2019: Portable, Reproducible High Performance Computing In the Cloud

- PEARC 2019: Portable, Reproducible High Performance Computing In the Cloud

Nathan Freeman is an Engineering Scientist Associate in the Cloud and Interactive Computing (CIC) group at the Texas Advanced Computing Center at the University of Texas at Austin. Nathan manages the development of the Tapis Workflows API and related services, libraries, and UI.
Previous trainings:

- PEARC 23: Best Practices of CI/CD for High Performance Computation with Tapis Workflows API
- TACCster 2022: Tapis day at TACC

Dr. Anagha Jamthe is a Research Associate in the Cloud and Interactive Computing group (CIC) at the Texas Advanced Computing Center at the University of Texas at Austin. Dr. Jamthe has played a key role in developing and validating Tapis APIs.
Previous trainings:

- PEARC 23: Best Practices of CI/CD for High Performance Computation with Tapis Workflows API
- TACCster 2022: Tapis day at TACC
- PEARC 22: Building Portable, Scalable and Reproducible Scientific Workloads across Cloud and HPC for Gateways
- Gateways 21: Portable, Scalable, and Reproducible Scientific Computing: from Cloud to HPC
- PEARC 2020: Leveraging Tapis For Portable, Reproducible High Performance Computing In the Cloud
- Gateways 2019: Portable, Reproducible High Performance Computing In the Cloud
- PEARC 2019: Portable, Reproducible High Performance Computing In the Cloud

Christian Garcia is an Engineering Associate in the Cloud and Interactive Computing group (CIC) at the Texas Advanced Computing Center at the University of Texas at Austin. Christian manages the Tapis Pods Service and related services, libraries, and UI.

Dhanny Indrakusuma is a Research Assistant in the Cloud and Interactive Computing group (CIC) at the Texas Advanced Computing Center at the University of Texas at Austin. Dhanny manages the Tapis ML Hub and related services, libraries and UI.

Resources:

1) Tapis Project: https://tapis-project.org
2) Tapis v3 documentation: https://tapis.readthedocs.io/en/latest/
3) Tapis Slack http://bit.ly/join-tapis

FABRIC Introduction and Tutorial

Paul Ruth
RENCI - UNC Chapel Hill
pruth@renci.org

James Griffioen
University of Kentucky
griff@netlab.uky.edu

Kuangching Wang
Clemson University
kwang@clemson.edu

***Abstract*—FABRIC project is an NSF Midscale research and education infrastructure that enables experimentation, rapid prototyping, and validation of new network and distributed computing methods and applications that are impossible or impracticable with commonly available research infrastructures.**

This tutorial will introduce attendees to the FABRIC testbed, sign them up with an account, and walk them through a hands-on experience with introductory and intermediate example experiments. Example topics will include: 1) Creating and deploying basic experiments, 2) Using wide-area networks, and 3) Using FABRIC's integrated measurement framework.

***Index Terms*—networking, testbed, tutorial**

I. Motivation

There is a broad range of infrastructure available to the research community, however the awareness of the community of the existence and capabilities of these testbeds remains limited. In fact, for many types of research these testbeds provided the resources and the flexibility to build scalable documented experiments that can then be shared and built upon by others. Importantly many of these platforms, namely FABRIC provide a path to scaling and evolving simple experiments into something that is persistent and can be used by external users, bridging the important valley that lies between a simple prototype and a robust system used by others. This effect where experiments can rapidly grow from simple to complex and become usable and useful systems can have a deep transformative effect on systems and networking research in the immediate future.

FABRIC testbed [Baldin et al.(2019)] was built with many of these ideas in mind - it provides the scale, robustness and programmability to meet these challenges. Even more importantly, FABRIC is designed from the ground up to be a 'fabric' that interconnects other experimental and production Cyberinfrastructure (CI) facilities around the world (Figure 1). Thus it achieves a multiplier effect of not only allowing the experimenters to harness its own resources, but to bring to bear the resources and investments in CI in many other disciplines thus vastly increasing the scale and reach of the experiments conducted on it.

II. Tutorial Organization and Content

The goal of this tutorial is to introduce participants to the FABRIC platform, its tools and resources and demonstrate how a simple experiment can be transformed into something complex, measurable and shareable. In the first session the attendees will get a basic introduction to FABRIC and a simple "Hello FABRIC!" experiment. In the second session they will proceed to grow this experiment to spans FABRIC's geographically distributed resources and employ FABRIC's measurement framework.

A. Sessions and Length

Session 1 (90 mins):

- **Introduction to FABRIC** (45 mins) - this lecture will describe FABRIC and its capabilities and provide a tour of the portal 2, Jupyter Hub, and documentation.
- **Hello, FABRIC** (45 mins) - in this hands-on component, participants will create FABRIC accounts and run a "Hello FABRIC!" experiment that demonstrates the basic features of the APIs

Session 2 (90 mins):

- **Intermediate Networking Experiment** (45 mins) - in this hands-on component, participants will work under the guidance of tutorial instructors to create a more complex experiment. In this experiment the participants will create a topology across multiple FABRIC locations, deploy a packaged networking experiments, and collect network performance data.
- **FABRIC Measurement Framework** (45 mins) - in this hand-on component, participants will deploy an experiment that utilizes the FABRIC measurement framework to collect low-level performance metrics from their experimental topology.

B. Recommended skill level

Minimum skill set: This tutorial is designed for users who have little or no experience with FABRIC. Basic experience with programming (i.e. python) and an ability to use remote virtual machines is required.

C. Technology and Software Requirements

All hands-on tutorials will use the FABRIC JupyterHub 3. Participants will need a laptop with Internet connectivity, a modern browser, and an SSH client. Specific instructions can be sent in advance for participants to create an account on the research infrastructure and be added to the designated project by the tutorial instructors.

Identify applicable funding agency here. If none, delete this.

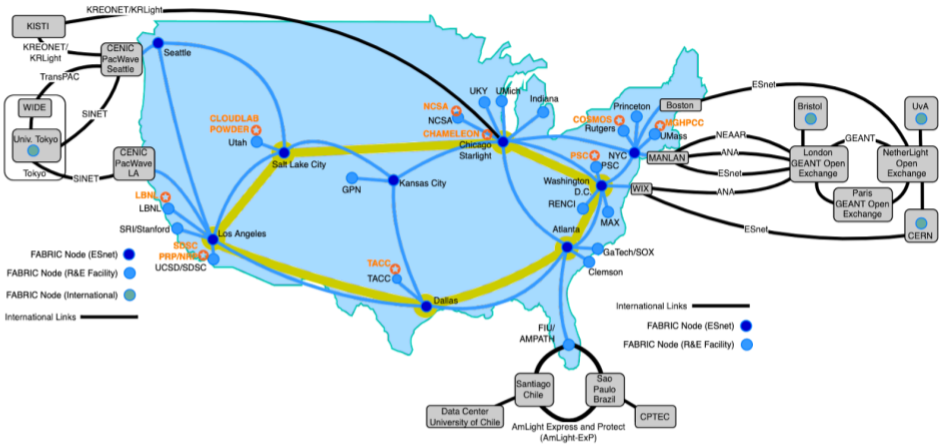


Fig. 1. FABRIC Infrastructure spanning the U.S., Europe, and Japan

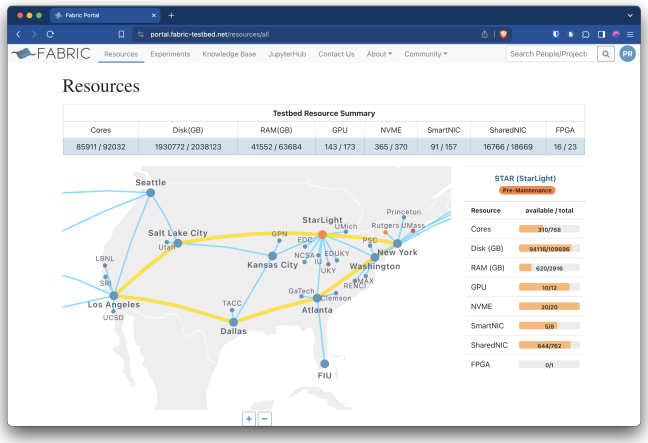


Fig. 2. FABRIC Portal

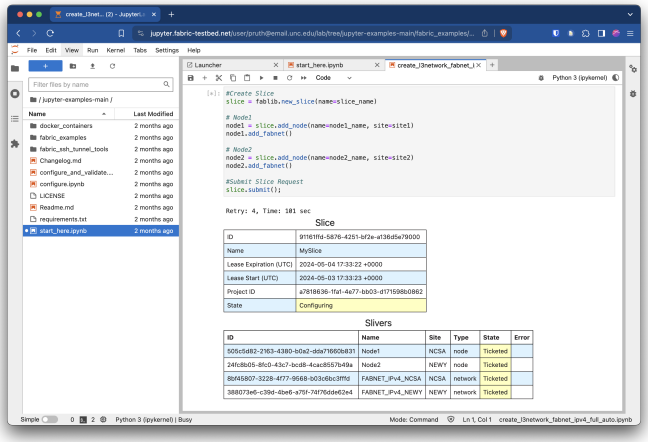


Fig. 3. FABRIC JupyterHub

III. References to previous iterations of similar tutorials

FABRIC team has presented introductory and advanced tutorials covering the GENI, Chameleon, and FABRIC testbeds at various venues over more than a decade. These venues include many of the GENI Engineering Conferences (GECs), Chameleon user meetings, FABRIC KNIT Workshops and other conferences. Recent examples include Internet2 TechEx [Ruth and Ilya(2022)], Massachusetts Open Cloud Alliance Workshop [Ruth(2023c)], SIGCOMM 2023 [Ruth(2023b)], and several FABRIC KNIT workshops [Ruth(2022)], [fab(2023)], [Ruth(2023a)], [Ruth(2024)].

Acknowledgment

This work is funded by NSF awards CNS-1935966, CNS-2029176, CNS-2029200, CNS-2029235, CNS-2029260, CNS-2029261, CNS-2330891

References

[fab(2023)] 2023. FABRIC KNIT6. https://learn.fabric-testbed.net/knowledge-base/knit-6-a-fabric-community-workshop/.

[Baldin et al.(2019)] Ilya Baldin, Anita Nikolich, James Griffioen, Indermohan Inder S. Monga, Kuang-Ching Wang, Tom Lehman, and Paul Ruth. 2019. FABRIC: A National-Scale Programmable Experimental Network Infrastructure. *IEEE Internet Computing* 23, 6 (2019), 38–47. https://doi.org/10.1109/MIC.2019.2958545

[Ruth(2022)] Paul Ruth. 2022. FABRIC KNIT5, FABRIC: Introduction and Advanced Tutorials. https://fabric-testbed.net/events/2022-knit-5-event.

[Ruth(2023a)] Paul Ruth. 2023a. FABRIC KNIT7. https://learn.fabric-testbed.net/knowledge-base/knit-7-a-fabric-community-workshop/.

[Ruth(2023b)] Paul Ruth. 2023b. FABRIC KNIT8. https://conferences.sigcomm.org/sigcomm/2023/tutorial-fabric.html.

[Ruth(2023c)] Paul Ruth. 2023c. Massachusetts Open Cloud Alliance Workshop, FABRIC Tutorial. https://massopen.cloud/2023-workshop/.

[Ruth(2024)] Paul Ruth. 2024. FABRIC KNIT8. https://learn.fabric-testbed.net/knowledge-base/knit-8-a-fabric-community-workshop/.

[Ruth and Ilya(2022)] Paul Ruth and Baldin Ilya. 2022. Internet2 TechEx: FABRIC Early Experimenter Tutorial. https://internet2.edu/2022-technology-exchange/tutorials/.

The Event Horizon Telescope Science Gateway - Progress and Lessons Learned

Rob Quick, Esen Gokpinar Shelton, and Jun Wang
Cyberinfrastructure Integration Research Center
Indiana University
Bloomington, Indiana
rquick@iu.edu ORCID: 0000-0002-0994-728X
egokpina@iu.edu ORCID:0000-0001-9520-0037
wang208@iu.edu ORCID: 0000-0002-8457-1235

***Abstract*—The Event Horizon Telescope (EHT) recently used 10 petabyte-scale observation data to construct the first images of black holes and 100 terabyte-scale simulation data to constrain the plasma properties around supermassive black holes. This work leveraged the Open Science Grid (OSG) high throughput resources provided by the Partnership to Advance Throughput Computing (PATh). While EHT has successfully utilized PATh to create the most extensive black hole simulation library to date, the broad adoption of this resource for data processing has been slower. The sophisticated command-line-driven HTCondor environment creates barriers for less technical researchers, limiting PATh's reach and impact on the broader astronomy and science communities. The Cyberinfrastructure Integration Research Center (CIRC) at Indiana University was awarded an NSF EAGER award to collaborate with EHT and PATh to implement a targeted science gateway instance that integrates critical EHT workflows to leverage OSG within the Apache Airavata framework. This work began in July of 2023. The EHT Science Gateway project was introduced to the Gateways community at last year's Gateways 2023 event. This paper and presentation will detail the work during the project's design, development, implementation, and release phases. The production release is scheduled for July 2024.**

***Index Terms*—High Throughput Computing, User Experience, Gateways for Science, Astrophysics, Apache Airavata**

I. Introduction

The Partnership to Advance Throughput Computing (PATh) is the NSF's premier resource for high throughput computing (HTC), delivering over 139 million core hours last year (one-year period ending 25-May 2024) to multi-institutional projects and an additional 274 million core hours to smaller groups that constitute the Open Science Grid (OSG) Connect community [1]. Usage groups include some of the most high-profile research investments by the NSF, such as the multi-institutional Event Horizon Telescope (EHT) collaboration. EHT used 10 petabyte–scale observation data to construct the first images of black holes and 100 terabyte-scale simulation data to constrain the plasma properties around supermassive black holes. While the EHT has successfully utilized PATh to create the biggest black hole simulation library to date, the broad adoption of this resource is much slower for data processing. The sophisticated OSG command-line-driven environment turns out to be a major barrier to the less technical users, limiting PATh's reach and impact within the wider astronomy and science communities.

National Science Foundation Award 2324672

Science gateways [2] have proven a successful mechanism for broadening access to scientific cyberinfrastructure (CI) by providing science-centric user environments tailored to end-user communities. An emerging and potentially transformative area of research in science gateway development involves the integration of established science gateway technologies with the methodologies of human-computer interactions (HCI) and user experience–based (UX) design.

To overcome barriers to adoption and create a highly functional science gateway environment that can effectively utilize the computing power of HTC resources, this project, which began in July 2023, will focus on the EHT community's use of PATh as a case study for this methodological approach. This exploratory project aims to demonstrate the effectiveness of combining human-computer interaction design methodologies with science gateway technologies, thereby enhancing the EHT's analysis of large astronomical data sets. This novel approach has the potential to transform both astrophysics and high throughput computing, offering a fresh perspective that benefits existing PATh user communities and supports the growth of PATh to include new research teams and projects.

Our team consists of experienced leaders from the EHT, HTC, and science gateways communities, bringing together a wealth of expertise in coupling CI resources with science stakeholder workflows and possessing in-depth knowledge of the EHT ecosystem. Working alongside these leaders are a science gateway developer and a graduate student from the Human-Centered Computing Department at Indiana University as well as a postdoctoral researcher in astronomy from the University of Arizona. Together, we have designed, prototyped, tested, and will release the EHT science gateway in July of 2024. The resulting science gateway will be highly functional and designed to meet the HTC needs for researchers ranging from experienced researchers to students. By prioritizing UX design and emphasizing the necessary human-computer interactions within a science gateway environment, we have

ensured a positive experience for EHT researchers, fostering continued engagement and reliance on PATh HTC resources.

While this project is short-term and focuses on the EHT collaboration, the foundational work of integrating science gateways with PATh-provided resources has the potential for long-term benefits for both the science gateway and PATh communities. Science gateways will enhance their toolbox of offerings by incorporating high-capacity, high-availability resources. PATh will get a UX-centric access method, providing an entry point for users unfamiliar with command-line interfaces and with minimal time or effort available to learn new technologies on their path to discovery.

II. EHT Science Gateway Prototype

In 2022, CIRC collaborated with EHT researchers from the University of Arizona to develop a proof of concept for the EHT gateway [3] as part of the XSEDE Extended Collaborative Support Services program. The proof of concept version demonstrated the basic integration of PATh resources with Apache Airavata, but it lacked the advanced functionality required by a large number of EHT researchers and did not incorporate user requirements or UX design methods. The current project builds on the proof of concept by creating a highly functional science gateway. This gateway will cater to a large portion of the EHT community and accommodate various levels of analysis tasks.

III. Project Objectives

This project is divided into three overarching objectives.

1) Deliver a highly functional, UX-centric gateway to the EHT project. This gateway will enable the integration of existing workflows that leverage OSPool resources.
2) Address the technical hurdles to allow the utilization of HTC resources from a science gateway environment within the Apache Airavata framework.
3) Conduct targeted outreach efforts aimed at astronomy researchers and the CI community to amplify the impact of the project.

IV. The Event Horizon Telescope Gateway

The prototype EHT gateway described previously allowed single runs of the ipole application on PATh resources and was demonstrated in a collaboration-wide training webinar. The proposed work will extend this prototype to launch large-scale image analysis from science gateways on PATh. We have focused on the most widely used EHT application, ipole, which provides spacial and numerical analysis of black hole images from the eight ground-based radio telescopes which make up the EHT international collaboration.

V. User Experience Design (UXD)

Our approach to designing the platform for EHT scientists using OSG resources has followed the double-diamond framework [4], which comprises four stages: Discover, Define, Develop, and Deliver. Here's a detailed overview of our process at each stage and our current status:

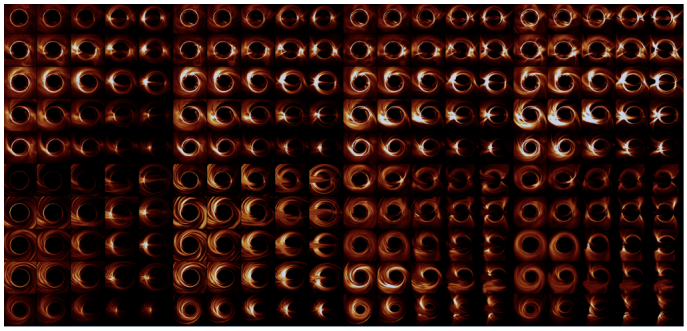


Fig. 1. Black hole simulation library generated for interpreting the image of the supermassive black hole at the center of the Milky Way. The computation was done on PATh-provided resources.

(1) Discover: In the Discover phase, we focused on understanding user needs. We immersed ourselves in the world of EHT scientists by engaging in conversations with stakeholders and analyzing user workflows. This exploration allowed us to identify pain points, user preferences, and specific challenges scientists face when using the OSG platform. By tracing the journey from job submission to the receipt of output files, we uncovered a crucial need for interactive analysis of output files and the pivotal role of post-processing in their research.

(2) Define: During the Define phase, we synthesized our findings from the Discover phase to pinpoint the core issues and user requirements. We revisited a previous proof of concept and conducted heuristic analyses to validate our initial insights. This helped us identify what worked well and areas needing improvement. By closely collaborating with two expert OSG users, we refined our understanding of user requirements. These interactions revealed the necessity for significant platform revisions, such as restructuring the platform to support multiple job submissions and improving information presentation for better clarity and ease of navigation.

(3) Develop: In the Develop phase, we transitioned from insights to solutions. Based on our research findings, we created three personas representing the diverse user types within the EHT community:

1) Expert Users with High Coding Skills: These users need a platform offering customization options and advanced scripting capabilities.
2) Novice Users, Early Researchers with Limited Coding Skills: These users seek user-friendly interfaces and intuitive workflows to simplify the submission process.
3) Students: This group includes users who utilize the platform for educational purposes.

We used these personas to guide our design decisions, ensuring the platform meets the unique needs of each user type. Additionally, we explored various use cases to anticipate a wide range of user interactions, enhancing the platform's adaptability and responsiveness. We mapped out the user journey from the homepage to job submission completion, detailing each step to ensure a seamless user experience.

(4) Deliver: We are currently in the Deliver phase, where we implement and refine our solutions based on iterative testing and feedback. Users begin their journey at the homepage, where they can sign up or log in. Upon logging in, they are directed to a personalized dashboard providing an overview of active applications, job statuses, and recent submissions. Users then select their desired application and choose their preferred submission method, whether through a web form for novices or a Python script for experts. Post-submission, users can monitor job progress and access completed analyses directly from the dashboard.

To sum up, throughout the design process, we identified several key takeaways that have significantly informed our development approach and feature prioritization. One of the highest priorities for the UI was to submit jobs to the OSG as intuitively as possible. We recognized that many users lacked the technical expertise to navigate complex command-line interfaces, so streamlining this process became critical. In this regard, we used several techniques such as information icons and tooltips to provide users with on-demand guidance throughout the submission process. Information icons next to key fields offered detailed explanations of what was required without overwhelming users with technical jargon. Furthermore, we organized the job submission process into a clear, step-by-step sequence, ensuring that users were guided through each necessary action, reducing the risk of error or incomplete submissions. Additionally, the platform provides real-time validation and feedback if a submission field is incomplete or contains incorrect data, enabling users to correct mistakes before submission.

While the majority of the features were made available through the user-friendly UI, the outcome analysis was specifically left for Jupyter notebooks. This decision was made because Jupyter offers more flexibility for advanced users to perform detailed and customizable analysis. However, to accommodate users with less experience in Jupyter, we provide pre-prepared workbook templates that guide them through the analysis process. These templates reduce the need for in-depth knowledge of Jupyter while still offering powerful analytical capabilities.

By adhering to the double-diamond framework, we have systematically developed a user-centered platform that addresses the specific needs and challenges of EHT scientists, ensuring a robust and responsive user experience.

VI. Integration with Apache Airavata

Our project is implementing a web-based science gateway that leverages the open-source Apache Airavata [5] gateway framework. The Airavata Django Portal is a web interface to the Apache Airavata API implemented using the Python Django web framework. The Airavata Django Portal can be used as is for a full-featured web-based science gateway. It can also be customized through various plugins to add domain-specific functionality. The primary objective of EHT gateway is to provide users with a streamlined and user-friendly interface for interacting with PATh resources, eliminating the challenges associated with command-line interfaces for users familiar with browser-based interfaces.

By implementing an Apache Airavata–based gateway with a graphical user interface (GUI), we will significantly reduce the learning curve associated with command-line interfaces, enabling more researchers to utilize HTC resources effectively. The web interface will be designed to be user friendly, providing clear instructions and visual aids to guide users through the process. This will make the system more accessible to researchers with varying levels of technical proficiency. Web-based interfaces provided by Apache Airavata have proved highly effective, allowing researchers to more quickly and efficiently utilize high performance computing (HPC) environments than traditional command-line interfaces. Moreover, the OSPool—accessible through Apache Airavata middleware—provides hundreds of thousands of hours of daily access to HTC resources, further enhancing its appeal to the scientific community.

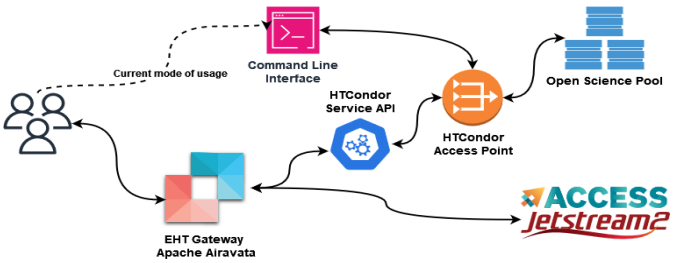


Fig. 2. A depiction of the proposed EHT science gateway, which will enable users to interact with Apache Airavata's gateway services and access the OSPool through HTCondor access points.

While coupling the Apache Airavata science gateway framework with PATh resources is an undoubtedly significant advancement and beneficial to both the HTC and Apache Airavata communities, it has taken substantial effort to fully integrate the two systems beyond a mere proof of concept. Our project will provide a highly integrated EHT portal while recording and addressing the challenges of interactions between web-based environments and HTC workload creation and submission. By addressing these challenges, we seek to streamline the adoption process for future projects that leverage these technologies. This, in turn, will enable the seamless utilization of HTC resources within science gateway environments.

In April of 2024, the first large-scale test run was submitted from the EHT gateway for OSG analysis. This test consisted of 3000 telescope images and considered six unique parameter conditions for 36000 HTC jobs. This run was completed in less than 24 hours. While some errors were encountered, more than 94% were completed successfully. By re-compiling iPole application with less aggressive optimization flags and increasing the value max_retries in submission, the error rates have reduced to less than 0.04% in the latest experiments.

The project has drawn upon the insights gained from PATh's EHT integration and concurrent survey work. This collaborative effort has established a strong foundation for supporting

additional research communities. The PATh project leadership, the science gateways community, and the project management are jointly identifying these communities.

VII. Training and Outreach Activities

The PATh team and the CIRC team at IU conducted joint tutorials throughout the 2022 calendar year. This included delivering an EHT-targeted webinar in September and conducting a tutorial at the Gateways 2022 conference. Building upon these events, we will further refine the materials and continue to provide training at CI events and target existing projects that could benefit from the collaborative PATh and CIRC relationship.

To expand our outreach efforts, upon production release of the EHT Science Gateway, the project team will specifically target EHT user groups at semiannual, in-person EHT collaboration meetings as well as through an online tutorial webinar event that we will organize. These outreach events feature tutorials focused on increasing the user base and identifying user needs for future functionality. In 2024, we plan to submit papers to the PEARC and the Science Gateways conference. We also hope to initiate engagements with two additional PATh-recommended communities currently using HTC resources but are likely to embrace science gateways during High Throughput Week in July 2024.

By strategically targeting these events and communities, we aim to enhance user engagement, expand the user base, and identify opportunities for collaboration and future development within the PATh framework.

A. Collaboration

The project receives dedicated effort and oversight from the funded institutions of Indiana University and the University of Arizona. The University of Illinois Urbana-Champaign, MIT Haystack Observatory, the Harvard-Smithsonian Center for Astrophysics, and the Harvard Black Hole Initiative will provide additional community technical and research support. To further strengthen the project, we will leverage the support and operational assistance of the PATh team from the University of Wisconsin–Madison and Morgridge Institute for Research. This collaboration entails hosting an OSPool Access Point and providing support for HTCondor integration issues that may arise throughout the project's duration.

Before the formal development process of the platform, informal consultations with some of these key stakeholders and potential collaborators were an integral part of the planning phase. These early conversations helped highlight some of the challenges and best practices learned from similar projects. As the project evolves, we plan to formalize this feedback loop with surveys and interviews to better capture insights and avoid pitfalls experienced by others in the astronomy community. By incorporating feedback from both internal users and external collaborators, we aim to ensure the gateway remains responsive to evolving user needs and adaptable to broader astronomical applications.

VIII. Project Deliverables

The project receives dedicated effort and oversight from the funded institutions of Indiana University and the University of Arizona. The University of Illinois Urbana-Champaign, MIT Haystack Observatory, the Harvard-Smithsonian Center for Astrophysics, and the Harvard Black Hole Initiative will provide additional community technical and research support. To further strengthen the project, we will leverage the support and operational assistance of the PATh team from the University of Wisconsin–Madison and Morgridge Institute for Research. This collaboration entails hosting an OSPool Access Point and providing support for HTCondor integration issues that may arise throughout the project.

IX. Collaboration

The principal project deliverable is a fully functional EHT Science Gateway. This gateway will support the EHT needs for preparing and submitting black hole image simulation (ipole) to PATh resources. A significant amount of effort has been concentrated on user experience (UX) design, allowing these applications to be widely usable by EHT scientists through science gateway user interfaces. We will integrate the lessons learned from the UX-based design and development of the EHT science gateway to integrate new science gateway portals that leverage HTC resources. These may include additional astronomy communities (such as the Vera Rubin Observatory), and other current OSPool users identified by our PATh collaborators. We will submit this work at both cyberinfrastructure-focused conferences (PEARC, High Throughput Week, Gateways) and to the multi-messenger astronomy community (Semi-annual EHT Collaboration Meetings). These outreach deliverables will include developer and user tutorials and training events.

References

[1] OSG by the Numbers, https://gracc.opensciencegrid.org/d/000000074/gracc-home?orgId=1

[2] Gesing, S., Wilkins-Diehr, N., Dahan, M., Lawrence, K., Zentner, M., Pierce, M., Hayden, L. and Marru, S., 2017. Science gateways: the long road to the birth of an institute.

[3] Event Horizon Telescope Prototype. Gateway https://eht.scigap.org/

[4] Kochanowska, M. and Gagliardi, W.R., 2022. The double diamond model: In pursuit of simplicity and flexibility. Perspectives on Design II: Research, Education and Practice, pp.19-32.

[5] Marru, S., Gunathilake, L., Herath, C., Tangchaisin, P., Pierce, M., Mattmann, C., Singh, R., Gunarathne, T., Chinthaka, E., Gardler, R. and Slominski, A., 2011, November. Apache Airavata: a framework for distributed applications and computational workflows. In Proceedings of the 2011 ACM workshop on Gateway computing environments (pp. 21-28).

A Serverless, Cost-Effective Hybrid Cloud Solution for Executing HPC Programs through Web Application Interfaces

Mark Durbin
Office of Research
University of Central Florida
Orlando, Florida
mark.durbin@ucf.edu

Nikhil Balachandra
Office of Research
University of Central Florida
Orlando, Florida
nikhil.b@ucf.edu

Paola Canales
Office of Research
University of Central Florida
Orlando, Florida
paola.canales-bigio@ucf.edu

Ezequiel Gioia
Office of Research
University of Central Florida
Orlando, Florida
ezequiel.gioia@ucf.edu

***Abstract*— This paper presents a serverless, cost-effective hybrid cloud solution for deploying web applications that leverage High-Performance Computing (HPC) resources. The proposed architecture addresses the challenges faced by researchers in securely exposing software running on HPC clusters to the internet while minimizing costs. By integrating AWS services with an on-premises HPC cluster, the solution enables researchers to deploy interactive web applications that offload computationally intensive tasks to the HPC cluster. The implementation details, including the technology stack and workflow, are discussed, highlighting the benefits of this hybrid approach.**

Keywords —hybrid-cloud, serverless-architecture, high-performance-computing, cloud-computing

I. Introduction

Researchers encounter challenges when showcasing resource intensive computer programs intended for high-performance computing clusters. These programs, characterized by custom dependencies and proprietary codebases, are impractical to install on typical desktop or laptop computers and often exceed the capabilities of consumer hardware. To address these limitations, researchers develop interactive solutions to execute their programs, hosting websites or applications directly on their high-performance computing clusters. These web applications enable users to run complex code through simple web interfaces, allowing a broader audience to access these tools without requiring specialized hardware or technical expertise. However, exposing these clusters directly to the internet raises security concerns and may breach institutional policies. Conversely, deploying self-contained solutions entirely in a public cloud environment can mitigate these security risks, but may be cost-prohibitive.

While these challenges are common in the field, our specific case presented a unique set of constraints that required a tailored approach. In our case, the researcher's requirements included maintaining code privacy, adhering to institutional policies, and operating within a limited budget while providing a user-friendly web interface for interacting with the HPC cluster.

These constraints made both fully cloud-based and exclusively on-premises approaches unworkable. A cloud-only solution was not viable due to budget constraints, as the costs of provisioning and maintaining cloud infrastructure would exceed the project's limits. Additionally, the need to keep the codebase private within the on-prem HPC cluster ruled out the use of publicly available science gateways. Deploying a dedicated on-prem web server or gateway was not feasible due to the lack of available infrastructure and the resources required for setup and maintenance. Furthermore, our solution needed to eliminate the need to open any inbound network ports on the HPC cluster, relying solely on outbound internet connections to maintain both security and simplicity while ensuring a seamless user experience.

To address these specific challenges and constraints, this paper presents a cost-effective hybrid cloud solution that combines the benefits of serverless cloud services and on-premises resources. By leveraging AWS services in conjunction with an on-premises HPC cluster, our proposed architecture enables researchers to securely deploy interactive web applications without the complexity of managing web servers. This approach minimizes costs by utilizing serverless technologies, maintains the privacy of the codebase by keeping it on-premises, and avoids direct exposure of the HPC cluster to the internet. The result is a streamlined solution that provides a web interface to interact with the HPC cluster while meeting all the identified requirements and overcoming the limitations of traditional approaches.

II. Solution Architecture

The proposed solution architecture integrates cloud services with an on-premises HPC cluster to provide researchers with a secure and cost-effective way to deploy interactive web applications. The architecture is composed of three key components: web application, job queue, and job queue worker.

The web application is deployed to managed hosting, with Continuous Integration and Continuous Deployment (CI/CD) from a GitHub repository. Users submit a form containing input parameters for the researcher's application, and these parameters are included in the message sent to the job queue. The job queue worker, running on the on-premises HPC cluster, continuously monitors the queue for new jobs, executes the researcher's application using the provided input parameters, stores output files in the cloud, publishes a message to the web application in real-time, and marks the job as completed. The solution leverages Python and JavaScript libraries and frameworks, such as Boto3, Node.js, and React.js, to create a cohesive and efficient serverless architecture.

III. Implementation Details

A. Web Application

The web application is hosted on AWS Amplify, a service designed for deploying full-stack web and mobile applications without the need to manage underlying infrastructure. The application displays a form in the user's browser, allowing them to specify input parameters for the research application. Since the app needs to be publicly accessible on the internet, traditional authentication and authorization mechanisms were not implemented. Instead, Google reCAPTCHA is used to prevent abusive form submissions. Once the form inputs are validated, the reCAPTCHA verified, and the form submitted, an API request is executed.

B. Job Queue

The job submission process is handled by a Representational State Transfer (REST) API developed with Amazon API Gateway and AWS Lambda. API Gateway receives the HTTP request and invokes a Lambda function that processes the form data and places a message in AWS SQS, a message queue service. The message represents a pending job for the HPC cluster and contains the user's specified parameters.

C. Job Queue Worker

Pending jobs remain in the message queue until they are polled and processed. A Python script that functions as a worker or job queue consumer, monitors for new jobs submitted into the SQS queue using HTTP long-polling. The polling process creates continuous loops of requests and responses, which allows the client to receive job submission updates. Long-polling reduces the number of requests made, reducing network traffic and server load, and minimizing computational costs. When a new job is detected by the worker, the worker processes the input data and uses it to execute a local command on the HPC cluster, for example, the instruction to execute a FORTRAN program. Once the program performs the computation and produces its output, the worker captures the program's response and uploads files the program generates to cloud object storage, Amazon S3.

D. Real-time Notifications

Providing real-time feedback after the web application form is submitted presents challenges due to the limitations of the

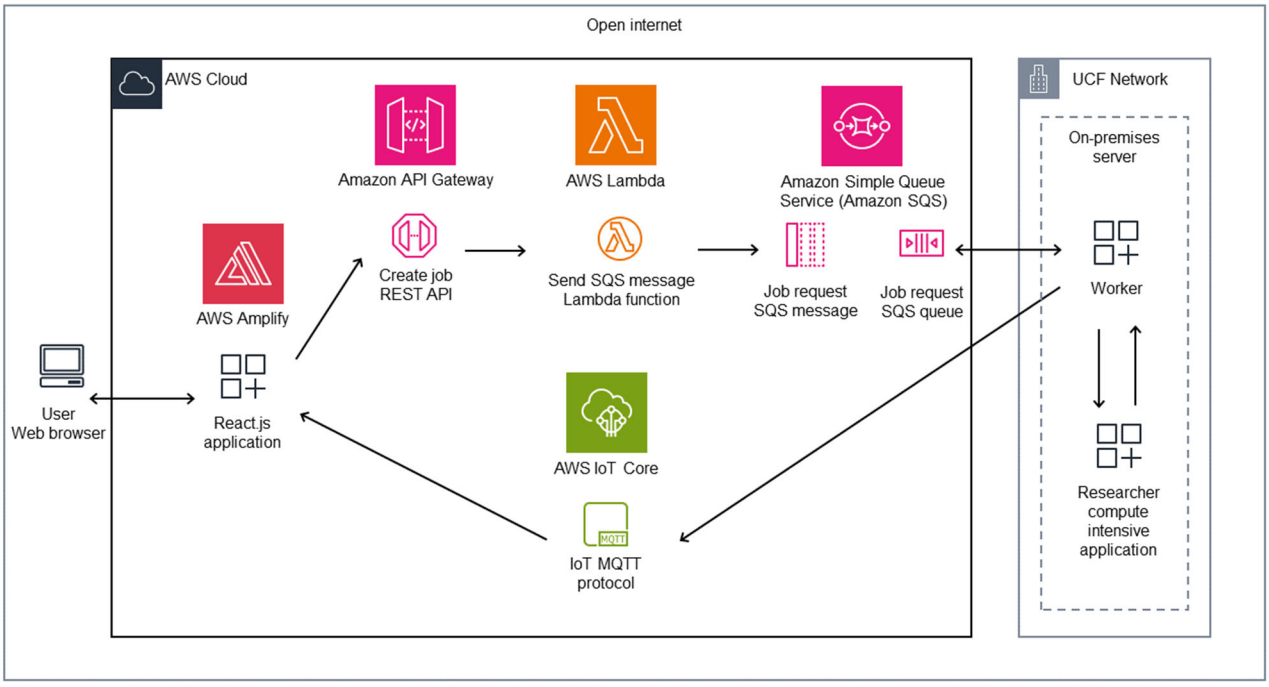


Fig. 1. Architecture diagram of the proposed solution

HTTP protocol, where requests are always initiated by the client. One approach to overcome this limitation is to make continuous requests to a REST API at consistent intervals to check the job status. While feasible, this solution is inefficient and requires developing a web API that will allow a client to make numerous and potentially dubious requests. Another approach is to utilize web sockets to initiate two-way communication, enabling the server to push updates back to the client. However, this would require a web server, that provides a web socket service, something that was not within the constraints of this project.

To address this challenge, the proposed solution leverages AWS IoT Core, a serverless message broker service designed for IoT device communication. AWS IoT Core establishes two-way communication through the MQTT protocol, treating each web client and the HPC cluster as IoT devices. This approach enables communication in both directions without the need to deploy additional infrastructure. AWS IoT Core employs a publish/subscribe (pub/sub) model, using "topics" to categorize and route messages between devices and applications. Publishers, such as devices or applications, send messages to the AWS IoT Core message broker on specific topics. Subscribers, also devices or applications, receive messages from the broker by subscribing to specific topics. This pub/sub model enables efficient, scalable, and flexible communication between IoT devices and applications.

In the proposed workflow, after submitting a job, the web client subscribes to a topic in AWS IoT Core. The topic ID is a globally unique object identifier (GUID) to ensure that the client only receives notifications related to their specific job, preventing interference with other users' jobs. Upon job completion, the worker publishes a JSON message to the AWS IoT MQTT endpoint and topic. This message contains the program's response, and secured temporary URLs, allowing the client to download the generated files. The web client, having subscribed to the relevant topic, receives the JSON message and updates the user interface accordingly. This approach provides near real-time feedback, including job status and links to download the output files, without the need for continuous polling or the deployment of a dedicated server for web socket communication.

E. Deployment

The deployment process for this system is handled through the AWS Amplify CLI, a command-line tool that automates the creation and configuration of cloud resources. It utilizes AWS' Infrastructure as Code tools, such as AWS CloudFormation and the AWS Cloud Development Kit (CDK), to generate the required configurations. After the initial setup, developers can use the Amplify “push” command to deploy the latest version of the web application along with any new or updated cloud resources. This command manages the deployment of the web application code, configures services like Lambda functions and API Gateway endpoints, and handles their configurations. However, the queue worker Python script, which interacts with these cloud services, must be manually copied to the HPC cluster to ensure proper integration with the newly deployed or updated cloud resources.

IV. BUDGET CONSIDERATIONS

In determining the most cost-effective solution for our computational workflow, we compared the costs of different infrastructure setups, reference Table III. While science gateways offer valuable tools for many research projects, they typically require a dedicated server. For example, running a web server on a virtual machine (t3.medium EC2 instance) on AWS with storage (50 GB) and a public IP address would cost approximately $29.55 per month. This example EC2 virtual machine is with a Compute Savings Plan of 1 year commitment with no upfront costs. This setup represents an average scenario in terms of minimizing costs. All AWS costs presented on this paper were calculated using the North Virginia (us-east-1) AWS region.

The web application component of the hybrid proposed solution is estimated to incur only $2.75 per month for 25,000 jobs. This includes AWS Amplify for the web hosting, AWS Lambda and Amazon API Gateway for the web API, Amazon Simple Queue Service for application integration, and AWS IoT Core as a pub/sub message broker. Please note, AWS provides a free tier for certain services, including AWS Lambda and Amazon SQS, which offer up to 1,000,000 requests per month at no cost. The cost estimate is based on 25,000 requests per month, well within this free tier limit.

TABLE I. SOLUTION COST FOR 25,000 JOBS PER MONTH

Resource	***Cost***
Amazon Simple Queue Service (SQS)	$ 0.00
AWS IoT Core	$ 0.04
AWS Lambda	$ 0.00
Amazon API Gateway	$ 0.09
AWS Amplify	$ 2.62
Total Cost	**$ 2.75**

The key driver in the total cost being AWS Amplify's web hosting service which is 95% of the estimated cost. What influences the cost for the Amplify service is determined by metrics such as total build time, amount of data stored, and amount of data served.

TABLE II. AWS AMPLIFY MONTHLY COST

Resource	***Usage***	***Cost***
Build time	100 minutes	$1.00
Data Stored	5 GB	$0.12
Data Served	10 GB	$1.50
Total Cost		**$ 2.62**

This hybrid solution offers a highly cost-effective alternative, with a total estimated monthly expense of just $2.75 for handling 25,000 monthly jobs. This approach leverages the free tier quotas offered by some of the AWS services, such as

AWS Lambda and Amazon SQS, and provides substantial savings and flexibility to accommodate future growth.

TABLE III. COMPARING SCENARIOS

Scenario	***Cost per Month***	***Cost per Year***
Web Server: *EC2 instance, t3.medium, 50 GB storage*	$29.55	$354.60
Amplify: *25,000 jobs per month*	$2.75	$33.00
Amplify: *1,000,000 jobs per month*	$12.10	$145.20

V. FUTURE WORK

Future work on the proposed hybrid cloud solution will focus on enhancing its capabilities and usability. This could involve integrating additional AWS services for improved monitoring and maintainability, as well as exploring containerization and serverless computing for increased scalability and portability. Seeking community feedback could help identify potential integration opportunities with existing tools such as the GenApp framework. Updating the front-end tooling from the deprecated standard React.js tooling (Create React App) to Next.js could offer greater flexibility in the application design, as well as long-term support from the open-source community. Finally, transforming this solution into a no-code service will allow researchers to showcase their HPC-based projects, deploying and managing HPC-powered web applications without the need for extensive programming knowledge.

VI. CONCLUSION

The implemented hybrid cloud solution successfully addresses the challenges faced by researchers in deploying web applications that leverage privately accessible HPC resources. By combining serverless cloud services with on-premises HPC clusters, this approach offers significant advantages. Managed serverless services incur minimal costs for this specific use case while eliminating the need for server patching, infrastructure management, and opening network ports. This simplifies deployment, enhances security, and reduces maintenance overhead.

This solution offers researchers a secure, cost-effective, and efficient method to disseminate and execute computationally intensive programs via intuitive interfaces. By enhancing security, ease of use, and significantly reducing operational costs, this model presents a scalable solution for future research needs without the complexity of managing additional infrastructure.

ACKNOWLEDGMENTS

We would like to thank the researcher, Dr. Shengli Zou, at the University of Central Florida (UCF) for bringing this challenge to our attention and collaborating with the UCF Research IT team to develop and implement this solution. We also acknowledge the support provided by AWS in terms of services and documentation.

REFERENCES

[1] *"GenApp"*, GenApp Framework. [Online]. Available: https://genapp.rocks/ [Accessed: 2023]

[2] *"React"*, React. [Online]. Available: https://react.dev/ [Accessed: 2023]

[3] *"Amplify documentation"*, AWS Amplify Gen 2 Documentation. [Online] https://docs.amplify.aws/ [Accessed: 2023]

[4] "Boto3 documentation", Boto3 1.34.112 documentation. [Online] https://boto3.amazonaws.com/v1/documentation/api/latest/index.html [Accessed 2023]

[5] *"What is ReCAPTCHA?"*, Google. [Online]. Available: https://www.google.com/recaptcha/about/ [Accessed 2023]

[6] *"MQTT: The Standard for IoT Messaging"*, MQTT. [Online] Available: https://mqtt.org/ [Accessed 2023]

[7] *"WebSocket API"*, Websocket.org. [Online]. Available: https://websocket.org/reference/websocket-api/ [Accessed 2023]

Tapis Machine Learning Hub Service for Science Gateways

Dhanny Indrakusuma, Joe Stubbs, Nathan Freeman and Anagha Jamthe
Texas Advanced Computing Center
The University of Texas at Austin
Austin, TX, USA
Email: dhannywi@utexas.edu, (jstubbs, nfreeman, ajamthe)@tacc.utexas.edu

***Abstract*—The adaptation of machine learning (ML) in scientific and medical research in recent years has heralded a new era of innovation, catalyzing breakthroughs that were once deemed unattainable. In this paper, we present the Machine Learning Hub (ML Hub) – a web application offering a single point of access to pre-trained ML models and datasets, catering to users across varying expertise levels. Built upon the NSF-funded Tapis v3 Application Programming Interface (API) and Tapis User Interface (TapisUI), the platform offers a user-friendly interface for model discovery, dataset exploration, and inference server deployment.**

***Index Terms*—machine learning, tapis, open-source, science gateways**

I. Introduction

In recent years, the integration of machine learning methodologies into scientific and medical research has propelled the boundaries of innovation, leading to remarkable breakthroughs. As highlighted in the 2024 Artificial Intelligence Index Report by Stanford University, this transformative adoption has ushered in a new era of possibilities [1].

In response to the evolving landscape of research, we designed Machine Learning Hub [2] as a user-friendly Tapis service [3] to help users with varying machine learning expertise discover and interact with pre-trained machine learning models and datasets. In addition to the details described below, we are working with the team at Intelligent Cyberinfrastructure with Computational Learning in the Environment (ICICLE) AI Institute [4] to create a federated models repository by including the machine learning models developed by researchers on the ML Commons platform into the ML Hub.

II. System Design

The Machine Learning Hub aims to streamline Tapis users' machine learning workflows. The system is divided into two distinct components: the user interface (UI) component and the server-side components. Communication between the server and the UI client is facilitated by the Flask-CORS extension [5] that handles Cross-Origin Resource Sharing (CORS), enabling cross-domain queries, and the authentication middleware uses Tapis API [6] to generate JSON Web Token (JWT) to authenticate a user's session. The server-side components consist of a set of RESTful APIs served over HTTP with Hugging Face Hub API [7] integration - consisting of the hub and the inference server, and persistent data storage. The server-side components are deployed to a Kubernetes cluster using Tapis Pods service [8]. We describe below the key functionalities of each component. An illustration of the ML Hub system can be seen in Fig. 1.

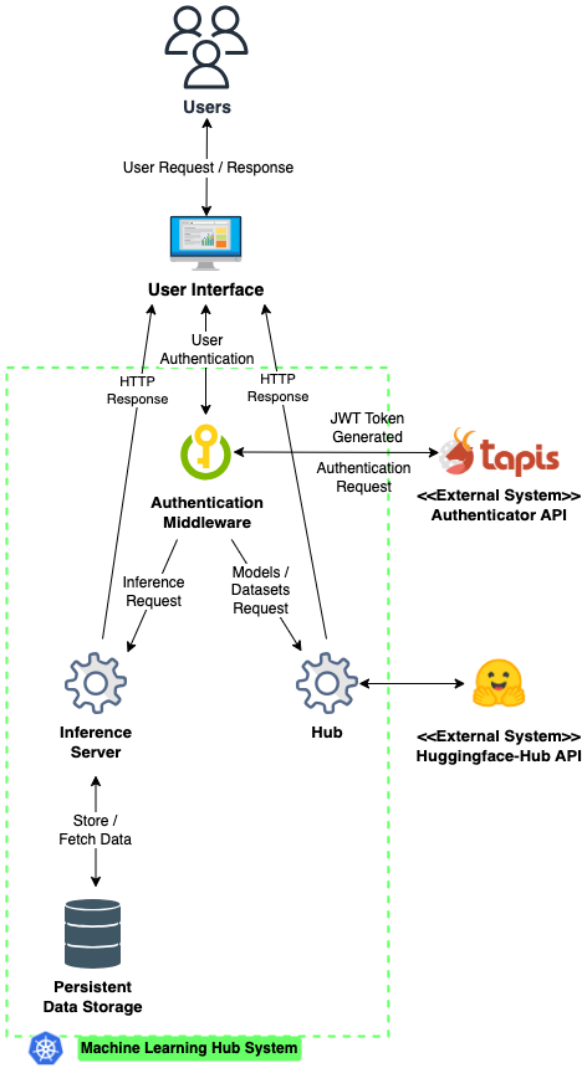


Fig. 1. Overview of ML Hub Service

A. Hub: Models and Datasets Discovery

The Hub consists of two REST APIs developed in Python Flask [9] - models hub and datasets hub. Each API has several endpoints with functionalities allowing users to view detailed popular models or datasets, filter them by specified parameters, view detailed information, and download models or datasets. We describe these functionalities in more detail below.

1) Models Hub: The models hub showcases the most downloaded machine learning models from Hugging Face [10]. In addition, it has additional functionalities such as (i) filtering models by author, task, trained dataset, query keyword, foundational libraries, and languages; (ii) fetching detailed information for a particular model and its associated model card; (iii) model download; (iv) checking the availability of inference server for a given model.

2) Datasets Hub: Like the models hub, the datasets hub allows users access to the top open-source datasets from Hugging Face and filter them by author, query keyword, task, language, size category, and official benchmark. In addition to fetching detailed information for a specified dataset and its dataset card, users also have the option to download the dataset.

B. Inference Server

Implemented using FastAPI [11], the inference server currently supports PyTorch-based [12] pre-trained models based on the FLAN-T5 (Text-to-Text Transfer Transformer) architecture [13]. The Flan-T5 is a fine-tuned T5 model that can perform various language tasks such as keyword generation and editing an English text based on the given instruction.

When running an inference, the inference server validates the user's request and then loads the user-specified model from the data storage. After processing the user's input, the server returns a JSON response containing the model's output. Users can also request that an inference server be provisioned if a model currently does not have an active inference server.

C. Persistent Data Storage

The persistent data storage uses a Network File System (NFS) volume to cache the machine learning models used by the inference server and the configuration file containing metadata of the associated models.

D. User Interface

The user interface (UI) for ML Hub is implemented using React [14] and TypeScript [15] and is currently under active development. The TypeScript types and fetch bindings for ML Hub are part of the @tapis/tapis-typescript NPM package [16]. The ML Hub module within the tapis-typescript is generated automatically from the Hub API's OpenAPI specifications [17], and the fetch bindings are then used to make API calls for the UI.

The UI for ML Hub is a service within TapisUI [18], a serverless web application built on the Tapis API. TapisUI runs entirely on GitHub pages and provides a user-friendly platform to interact with the models, datasets, and inference server. Screenshots and descriptions of the user interface can be seen in Fig. 2.

III. Target Users

The target users for this demo fall into two categories:

- Researchers with domain expertise that utilize national, campus, and local cyberinfrastructure resources who want to leverage machine learning to improve their research outcome but do not have machine learning expertise.
- Researchers with moderate machine learning expertise who are looking for a simpler way to host their model without worrying about user authentication and network issues.

IV. Session

We presented the Machine Learning Hub as a dynamic platform designed to help researchers with different levels of machine learning expertise discover and explore pre-trained machine learning models and datasets. For the session, we will begin by giving the audience a high-level overview of ML Hub functionalities and proceed with a live demo.

During the live demo, we will show how a user can utilize the ML Hub service within TapisUI to discover models and datasets, as well as interact with the inference server to run inference and initiate a model's inference server deployment. In case of a poor internet connection, we will present a pre-recorded version of the demo.

Acknowledgment

The Machine Learning Hub project is supported by the National Science Foundation Division of Advanced CyberInfrastructure Awards: 1931439, 1931575, and 2112606.

References

[1] R. Perrault and J. Clark, *Artificial Intelligence Index Report 2024*. Stanford University's Institute for Human-Centered Artificial Intelligence, 2024. [Online]. Available: https://aiindex.stanford.edu/wp-content/uploads/2024/04/HAI_2024_AI-Index-Report.pdf

[2] D. Indrakusuma, N. Freeman, and J. Stubbs, "Machine learning hub for tapis poster presentation," in *Science Gateways 2023 Annual Conference*. Zenodo, 2023. [Online]. Available: https://doi.org/10.5281/zenodo.10055681

[3] J. Stubbs, R. Cardone, M. Packard, A. Jamthe, S. Padhy, S. Terry, J. Looney, J. Meiring, S. Black, M. Dahan, S. Cleveland, and G. Jacobs, "Tapis: An api platform for reproducible, distributed computational research," in *Advances in Information and Communication*, K. Arai, Ed. Cham: Springer International Publishing, 2021, pp. 878–900.

[4] (2021) Icicle. Last access: 2024-05-31. [Online]. Available: https://icicle.osu.edu/

[5] (2013) Flask-cors. Last access: 2024-05-31. [Online]. Available: https://github.com/corydolphin/flask-cors

[6] (2020) Tapipy. Last access: 2024-03-31. [Online]. Available: https://github.com/tapis-project/tapipy

[7] (2020) Hugging face hub. Last access: 2024-03-08. [Online]. Available: https://github.com/huggingface/huggingface_hub

[8] R. C. Christian Garcia, Joe Stubbs and N. Freeman, "Tapis pods service exploration and initial performance analysis," in *Science Gateways 2023 Annual Conference*. Zenodo, 2023. [Online]. Available: https://doi.org/10.5281/zenodo.10034631

[9] (2023) Flask. Last access: 2023-4-19. [Online]. Available: flask.palletsprojects.com/en/2.3.x/

[10] (2017) Hugging face. Last access: 2024-03-08. [Online]. Available: https://huggingface.co

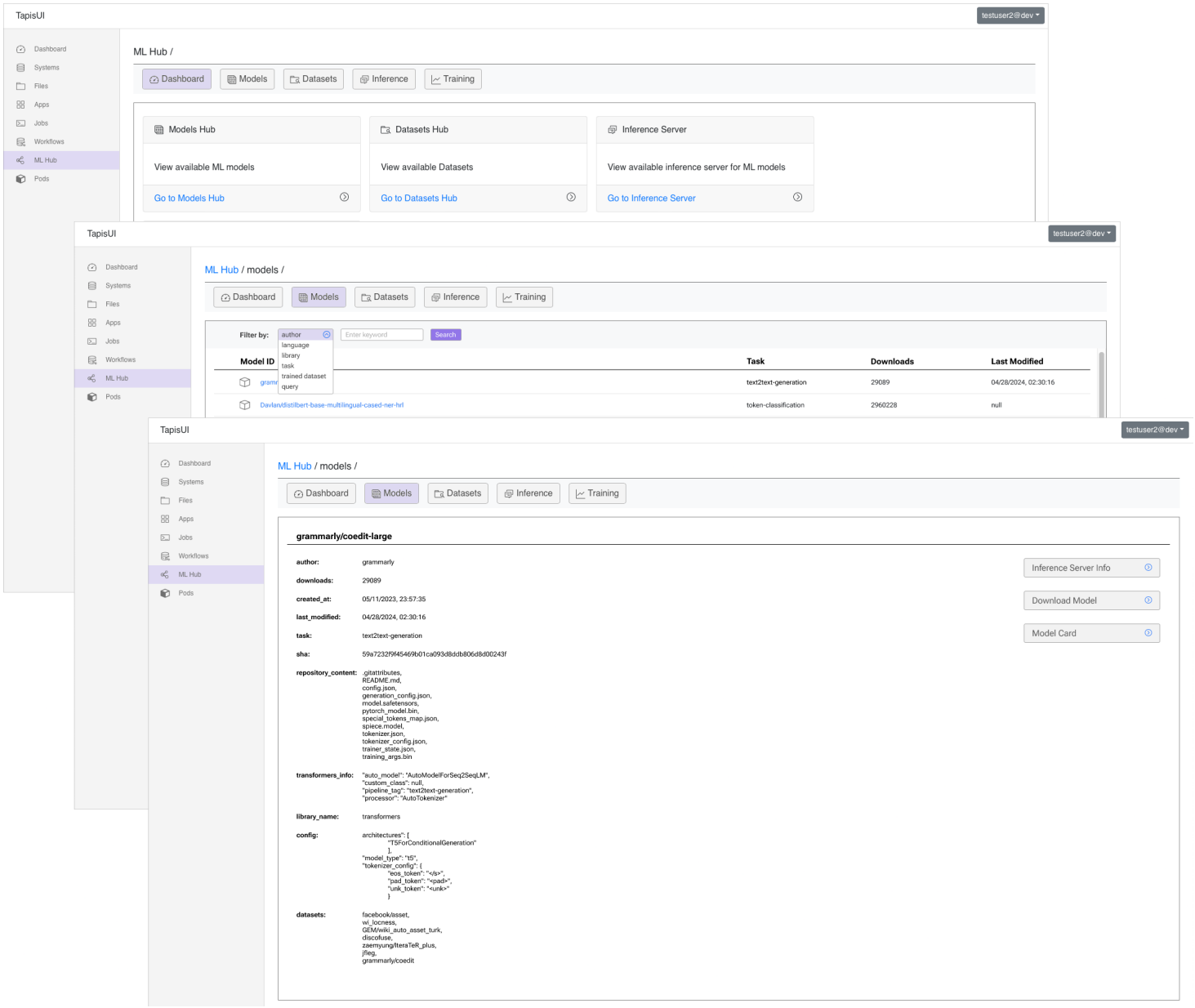


Fig. 2. User Interface for ML Hub, from top to bottom: (1) ML Hub Dashboard, (2) Top 100 models with filtering functionalities, and (3) A single model details page

[11] (2018) Fastapi. Last access: 2024-03-08. [Online]. Available: https://fastapi.tiangolo.com

[12] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[13] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. Dehghani, S. Brahma, A. Webson, S. S. Gu, Z. Dai, M. Suzgun, X. Chen, A. Chowdhery, S. Narang, G. Mishra, A. Yu, V. Zhao, Y. Huang, A. Dai, H. Yu, S. Petrov, E. H. Chi, J. Dean, J. Devlin, A. Roberts, D. Zhou, Q. V. Le, and J. Wei, "Scaling instruction-finetuned language models," 2022. [Online]. Available: https://arxiv.org/abs/2210.11416

[14] (2013) React. Last access: 2024-05-31. [Online]. Available: https://react.dev/

[15] (2012) Typescript. Last access: 2024-05-31. [Online]. Available: https://www.typescriptlang.org/

[16] (2021) Tapis-typescript. Last access: 2024-05-31. [Online]. Available: https://github.com/tapis-project/tapis-typescript

[17] (2017) Openapi specification. Last access: 2023-02-13. [Online]. Available: https://swagger.io/specification/

[18] J. Y. Chuah, J. Rosenberg, K. Strmiska, J. Stubbs, S. Cleveland, and J. McLean, "Tapis ui - a rapid deployment serverless science gateway built on the tapis api," in *Gateways 2021*. Zenodo, 2021. [Online]. Available: https://zenodo.org/record/5570569

Automating Instrument Data Analysis, Sharing and Publication Using the Globus Portal Framework

Vas Vasiliadis
University of Chicago
Chicago, IL, U.S.A.
0000-0002-4486-5431

***Abstract*—Managing the deluge of data from high-resolution microscopes, genome sequencers, and other research instruments is a growing burden for research computing professionals. Over the past seven years we have employed the modern research data portal (MRDP) design pattern to facilitate management of instrument-generated data, from creation through publication. In recent implementations of the design pattern, we incorporate data search, remote computation, and publication of resulting data products for downstream discovery, effectively making data FAIR (Findable, Accessible, Interoperable, Reusable) by default. We will demonstrate one such implementation and describe how it may be used by science gateway developers to jumpstart their own efforts.**

Keywords—instrument data management, data portal, remote computation, search, data discovery, mrdp.

I. Introduction

Since its inception in 2017, the modern research data portal (MRDP) design pattern [1] has helped research institutions that implemented it, dramatically improve performance when moving data. Beyond fast, reliable data transfer, data portals that adhere to the MRDP design pattern allow broad access through federated authentication, while maintaining fine-grained access controls. In many instances, data accessible via portals is retrieved and analyzed to support new research directions. Traditionally, this requires a series of disconnected actions, including (1) moving data to a remote compute resource, (2) creating the required environment in which to analyze the data, (3) starting the analysis and managing it through its completion, (4) moving the resulting data to a suitable system for publication, (5) defining or extracting metadata to describe the data, and (6) adding the metadata to an index so that it may be searched by others. We have developed a framework that automates these actions and removes the friction associated with working in multiple, distributed data and computation environments. A particular focus of this framework is to ensure that data are findable, accessible, interoperable, and reusable (FAIR) [2] without requiring complex, out-of-band-processes. The framework has been used to implement data portals and science gateways that effectively automate research operations for instruments at core facilities and national laboratories. We demonstrate here one such implementation that can be leveraged by developers building their own science gateways, data portals or data commons to support

II. Reference Implementation Description

Our implementation comprises the following components:

- A science DMZ [3] that eliminates the bottlenecks created by network configurations not designed for research. We refer the reader to the Energy Sciences Network website (fasterdata.es.net/science-dmz) for a comprehensive description of science DMZ concepts and components.
- Django (www.djangoproject.com), a Python framework for developing web applications. The reference implementation uses Django to implement template-driven search results and landing pages for the portal. It integrates with various Globus platform services to provide a streamlined user experience, from initial login through data analysis and publication.
- Globus Auth [4], a foundational federated authentication and access control service. The reference implementation enables users from over 1,800 organizations to access to the portal using their existing institutional credentials.
- The Globus Search service for managing scalable search indexes with and fine-grained access control over user-defined metadata. The reference implementation includes a multi-faceted index that allows the user to progressively refine a search query until the desired data are found.
- The Globus Compute service for managed remote computation using the function-as-a-service paradigm. Globus Compute was previously known as *func*X [5]. The reference implementation allows the user to analyze selected data on a remote computing resource. The portal is agnostic with respect to the compute resource, and allows analyses to be run at all scales, wherever the user has access to computing capacity.
- The Globus Flows service, for reliable orchestration of data management and compute tasks at scale. The reference implementation uses this service to enable "fire-and-forget" movement, analysis, and sharing of data without requiring human intervention. The portal uses the Flows service to initiate downstream processing of data returned by searches.

III. Demonstration

We will demonstrate the following user experience:

1) Authentication: Log into the science gateway using an existing institutional identity.

2) Data Discovery: Search and select a dataset of interest.

3) Data Analysis: Invoke a compute job on a remote system to analyze the selected dataset.

4) Results Sharing: Transfer the data to a system of the user's choice and make it available to collaborators.

Steps 3 and 4 above will be handled by an automated flow trigerred by the user.

REFERENCES

[1] K. Chard, E. Dart, I. Foster, D. Shifflett, S. Tuecke, J. Williams, "The Modern Research Data Portal: a design pattern for networked, data-intensive science," in PeerJ Computer Science, 2017, https://peerj.com/articles/cs-144.

[2] Wilkinson, M., Dumontier, M., Aalbersberg, I. *et al*, "The FAIR Guiding Principles for scientific data management and stewardship". Sci Data 3, 160018 (2016). https://doi.org/10.1038/sdata.2016.18

[3] E. Dart, L. Rotman, B. Tierney, M. Hester, J. Zurawski, "The Science DMZ: a network design pattern for data-intensive science," in SC '13: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, November 2013, Article No.: 85, Pages 1–10, https://doi.org/10.1145/2503210.2503245.

[4] S. Tuecke, R. Ananthakrishnan, K. Chard, M. Lidman, B. McCollam, S. Rosen, and I. Foster, "Globus Auth: A Research Identity and Access Management Platform," in IEEE 12th International Conference on eScience, 2016.

[5] Z. Li, R. Chard, Y. Babuji, B. Galewsky, T.Scluzacek, K. Nagaitsev, A. Woodward, B. Blaiszik, J. Bryan, D. Katz, I. Foster, K. Chard, "*funcX*: Federated Function as a Service for Science," in IEEE Transactions on Parallel and Distributed Systems, Volume: 33, Issue: 12, December 2022, pp. 4948-4963.

A Lidar and Radar Meteorology Science Gateway for Education and Research on the NSF Jetstream2 Cloud

1st Jennifer DeHart
Department of Atmospheric Science
Colorado State University
Fort Collins, CO USA
https://orcid.org/0000-0001-8536-4927

2nd Brenda Javornik
Earth Observing Laboratory
NSF NCAR, UCAR
Boulder, CO USA
brenda@ucar.edu

3rd Julien Chastang
NSF Unidata Program Center
UCP, UCAR
Boulder, CO USA
https://orcid.org/0000-0003-2482-3565

4th Ana Espinoza
NSF Unidata Program Center
UCP, UCAR
Boulder, CO USA
https://orcid.org/0000-0002-6292-073X

5th Michael Dixon
Earth Observing Laboratory
NSF NCAR, UCAR
Boulder, CO USA
https://orcid.org/0000-0002-9597-9113

6th Michael Bell
Department of Atmospheric Science
Colorado State University
Fort Collins, CO USA
https://orcid.org/0000-0002-0496-331X

***Abstract*—This paper introduces a lidar and radar meteorology science gateway deployed on the NSF Jetstream2 cloud, designed to enhance educational and research activities in atmospheric science. Utilizing the "Zero to JupyterHub with Kubernetes" workflow, we have created a science gateway that integrates lidar and radar meteorology software packages, notably the Lidar Radar Open Software Environment (LROSE). This integration allows users to execute applications directly from the JupyterLab terminal, streamlining the creation of datasets for further analysis and visualization within Jupyter notebooks. By combining traditional command-line operations with modern Python-based tools for data analysis and visualization, this gateway provides a robust end-to-end solution that caters to both educational and research needs. The gateway has already been vital in facilitating LROSE instructional workshops and will see future enhancements such as GPU acceleration to boost performance. Our work demonstrates the significant potential of merging established scientific computing techniques with advanced Python environments, opening new avenues for computational science education and research.**

***Index Terms*—LROSE, Cloud-based Science Gateway, Radar Meteorology, Lidar Meteorology, Data Visualization, Educational Technology in Atmospheric Science, JupyterHub, NSF Jetstream2, Atmospheric Data Analysis**

I. Introduction

In the rapidly evolving field of computational science, integrating traditional scientific tools with modern data analysis platforms is essential for advancing research and educational outcomes. The Lidar Radar Open Software Environment (LROSE; [1] [2] [3]) is an open-source project collaboratively developed by Colorado State University's Department of Atmospheric Science and the Earth Observing Laboratory at the National Center for Atmospheric Research. It provides modular tools for processing lidar and radar observations, including data conversion, derived quantities, quality control, gridding, and visualization. The integration of Jupyter notebooks and JupyterLab with sophisticated scientific computing tools like LROSE facilitates access to advanced analytical capabilities, allowing users to manipulate software packages directly from a web browser. By deploying these tools on the NSF Jetstream2 cloud [4] [5] through a JupyterHub science gateway, we significantly lower barriers to entry and democratize access to compute resources for educational institutions and researchers, facilitating seamless transitions between executing complex command-line tools and visualizing outputs in pre-configured Jupyter notebooks. Additionally, it facilitates the hosting of LROSE instructional workshops, enhancing learning through hands-on experience. This integration, which includes tools like the Python ARM Radar Toolkit (Py-ART) [6], enhances educational and research activities by making advanced lidar and radar data analysis more accessible.

A. LROSE Background

The goal of the LROSE suite is to provide the lidar and radar community with well-tested applications for scientific research. The primary programming language of most LROSE applications is C++, providing the computational speed to work with large datasets. The LROSE team contributes to the Open Source Radar community [7]. LROSE applications can be categorized into the following: data conversion, display, gridding, quality control, and scientific analysis of echo and Doppler wind observations. Examples of the scientific analysis supported by LROSE include particle identification (e.g., distinguishing between rain and snow), quantitative precipitation estimation analysis, and wind retrievals from either a single or multiple Doppler instruments. Most applications, particularly those that perform scientific analysis, are based on algorithms

that are described in peer-reviewed literature, of which RadxPid [8] and RadxEvad [9] are examples.

B. Why LROSE in a Science Gateway?

The LROSE team first explored the idea of an LROSE Science Gateway to improve accessibility. Educational exercises are often complicated by users having a variety of operating systems, potentially different versions of LROSE, and varying levels of experience with the applications. A cloud-based science gateway removes any installation barrier and ensures all participants are using the same version of the software. The use of JupyterHub as a science gateway was proposed by Zonca and Sinkovits [10]. Expanding upon the idea of using JupyterHub as a science gateway, since 2020 the NSF Unidata Program Center at the University Corporation for Atmospheric Research, has deployed numerous specialized PyAOS (Python for Atmospheric and Ocean Science) JupyterHubs tailored for semester-long academic courses and workshops [11] [12]. JupyterHub, and particularly Jupyter notebooks, is ideal as a Science Gateway for LROSE as notebooks mimic user workflows.

Previously, the LROSE team explored a specialized graphical user interface (GUI) as a gateway; however, any modifications to the workflow required development by software engineers. In general, the LROSE audience is familiar with programming in Python and wants to customize their interactions with LROSE. With Jupyter and Python, the interface is directly customized, without the need for additional software engineering. Specialized JupyterLabs can be deployed for specific workshops by the software engineers and radar scientists can focus their efforts on tutorial and Jupyter notebook materials. Finally, community members can develop their own workflows and submit them to the Gateway, thereby increasing engagement.

II. Methodology

In the spring of 2023, the LROSE team acquired an exploratory allocation on the NSF Jetstream2 cloud at Indiana University through ACCESS [5]. To develop the LROSE Science Gateway [13], we employed the "Zero to JupyterHub with Kubernetes" workflow ported to the NSF Jetstream2 cloud [10], enabling rapid and scalable deployment to accommodate a variable number of users. Authentication is managed via GitHub OAuth, reducing the maintenance burden. Since LROSE is a collection of C/C++ applications, we configured Docker containers based on the Jupyter Docker Stack to integrate the LROSE software, including command-line tools such as RadxConvert, FRACTL (Fast Reorder and CEDRIC Technique in LROSE), and SAMURAI (Spline Analysis at Mesoscale Utilizing Radar and Aircraft Instrumentation; [14]) available via the JupyterLab terminal. These containers also include Conda package manager environments equipped with Python packages like Py-ART [6] and Metpy [15] for further data analysis. A shared drive visible to all JupyterHub users contains instructional datasets for lidar and radar data analysis, which users can access and manipulate using LROSE command-line tools. Additionally, we crafted custom Jupyter notebooks that can function as tutorials or operational tools, complete with pre-loaded examples to guide users from data analysis to visualization.

A significant amount of effort was made to optimize resource allocation and enable the execution of computationally intensive command-line applications like SAMURAI within a multi-user JupyterHub environment. SAMURAI's substantial resource demands often caused the single user server to stall, rendering it inaccessible. Additionally, heavy concurrent usage by multiple users risked overloading the entire Kubernetes cluster. To mitigate these issues, we increased the CPU and RAM allocation per user and adjusted Kubernetes pod configurations to ensure the JupyterHub "core" pods are isolated from the single user pods which utilize the majority of CPU cycles when running such intensive workflows [16]. These adjustments aimed to ensure cluster stability, despite occasional individual node disruptions. This remains an active area of research as we continue to refine our resource management strategies.

III. Results and Discussion

A. Education Applications

These tutorials are designed to be used in workshop or classroom environments or by users who wish to learn more about specific LROSE tools. All the notebooks are also available on GitHub for offline usage on personal devices. The current tutorials offered on the LROSE Science Gateway highlight data quality control, particle identification (e.g., distinguishing between rain and snow, identifying ground clutter or birds), quantitative precipitation estimation, multi-Doppler analysis (e.g., SAMURAI), and the creation of mosaics from multiple radars. The user interface and the introductory portion of the quantitative precipitation estimation tutorial are shown in Fig. 1. Initial tutorials provided pre-made parameter files so a user could run all relevant applications without worrying about incorrect parameters that might produce errors. Based on user feedback, we have since created a "guided" multi-Doppler analysis tutorial that helps the user understand how to set each necessary parameter, inspired by instructional notebooks created for Metpy [15]. We plan to expand this type of tutorial to the existing notebooks.

Our first class-based tutorial took place in the spring of 2024, where students in Prof. Michael Bell's Radar Meteorology class generated dual-Doppler analyses from two mobile radars that sampled a tornado in Goshen County, Wyoming observed during the Verification of the Origins of Rotation in Tornadoes Experiment (VORTEX2) field campaign in 2009. LROSE applications RadxPrint, FRACTL, and SAMURAI were used to generate gridded analyses of the two-dimensional wind field. RadxPrint provides the students with the radar file metadata, including variable names needed to run FRACTL and SAMURAI. FRACTL and SAMURAI use two different approaches to retrieve two-dimensional winds from radar data; FRACTL's point error statistics help identify regions where the SAMURAI analysis is of higher quality. Once the gridded

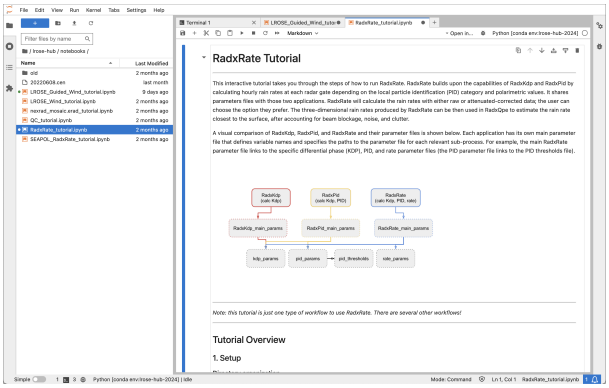


Fig. 1. *A screenshot of the JupyterLab layout and the RadxRate tutorial that a user sees after logging into the LROSE Science Gateway.*

analyses were produced, standard Python packages matplotlib [17], numpy [18], and xarray [19] were used in conjunction with cartopy [20] and Metpy [15] to visualize the wind field and storm structure; one of the images the students created in the tutorial is shown in Fig. 2, showing the horizontal tornado circulation in vectors and the radar reflectivity in color shading. Students provided positive feedback about the exercises in the "guided" tutorial and constructive feedback to improve tutorials for the future.

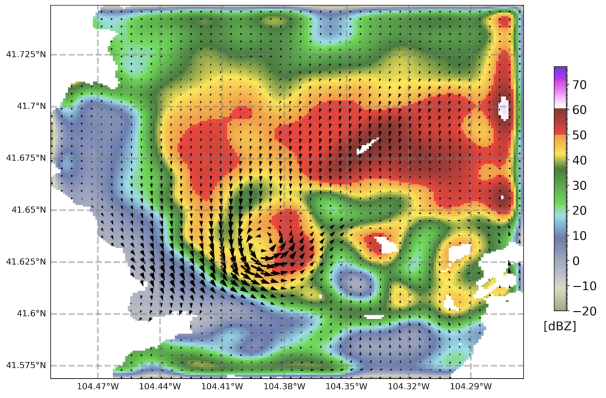


Fig. 2. *Map of SAMURAI-analyzed 1.5-km radar reflectivity (colors) and wind vectors (arrows) of the VORTEX2 tornado case analyzed during the Spring 2024 classroom exercise.*

B. Challenges

As previously discussed, computationally-intensive applications like SAMURAI are challenging to run, even when data sets are carefully chosen to minimize memory and computational usage for demonstrations and exercises. For example, SAMURAI will utilize as much CPU as is allocated to the environment–in this case, the Kubernetes pod–in which it is executed. When stress testing SAMURAI on the JupyterHub with two simultaneous JupyterLab instances on the same node, we noticed a slow down in the execution time as compared to the same workflow running on only one JupyterLab instance at a time. In addition, the step in the workflow where data is read into the program also suffers a significant performance hit with two simultaneous workflow executions. While the latter is being investigated in the SAMURAI code, the former is still a subject of research and uncertainty. It is speculated that the multiple degrees of separation from the physical hardware (i.e., a hypervisor/VM, Kubernetes/containerization, JupyterHub/Lab) may affect how efficiently SAMURAI and other computationally intensive processes are executed. We find it necessary to close the knowledge gap using new or existing documentation and research in order to better determine the cause of the performance losses due to simultaneous execution. One way to mitigate them is to isolate single user pods to their own Jetstream2 virtual machine. This solution, however, is only realistically viable for certain scenarios, as described below.

With open-ended assignments, in contrast to time-limited workshops, preallocating virtual machines can unnecessarily drain ACCESS resources, particularly when usage is inconsistent (e.g., overnight when students are not working). In the cases where gateway usage is better known, we use the soft scale technique to quickly bring nodes up and down as needed [21]. This technique reduces the allocation usage rate by allowing for worker nodes to be pre-configured as part of the Kubernetes cluster and then "shelved," a VM state that does not exhaust Service Units. However, this technique requires developer intervention. Automated scaling of virtual machines on Jetstream2 would be a useful functionality in the future. The authors are open to feedback or suggestions on how the hurdles described in this section can be overcome.

IV. Conclusion and Future Work

In conclusion, the LROSE Science Gateway enhances the accessibility and usability of advanced atmospheric analysis tools in education and research. By leveraging the NSF Jetstream2 cloud infrastructure and integrating it with JupyterHub, we have effectively reduced the entry barrier for students and researchers to utilize sophisticated lidar and radar data analysis tools. This gateway facilitates seamless transitions between command-line operations and interactive data visualization, making it easier to enter this specialized field.

Forthcoming development efforts will focus on improving the gateway's performance and exploring GPU acceleration. Future educational plans include partnering with more university classrooms, running an intermediate SAMURAI tutorial to help users understand how the advanced parameters affect scientific results, and developing more tutorials to encompass more aspects of lidar and radar meteorology. Based on the positive feedback to our "guided" multi-Doppler analysis tutorial, we plan to upgrade our remaining notebooks to teach users to create their own parameter files independently. New tutorials are in the preliminary stages of development, which would highlight other LROSE applications and explain how to debug common error messages. Finally, we intend to showcase how users can integrate LROSE applications with other open-source packages (e.g., Py-ART, CSU RadarTools [22]).

V. Acknowledgments

LROSE development is supported by NSF grants #2103776 and #2103785. This material is based upon work supported by the NSF National Center for Atmospheric Research, which is a major facility sponsored by the U.S. National Science Foundation under Cooperative Agreement No. 1852977. NSF Unidata Program Center funding is supported by NSF grant #1901712. This work used NSF Jetstream2 at Indiana University through allocation #EES200002 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, which is supported by National Science Foundation grants #2138259, #2138286, #2138307, #2137603, and #2138296.

VI. References

[1] M. M. Bell, "nsf-lrose/lrose-topaz: lrose-topaz-20220222." Zenodo, 2022. doi: 10.5281/zenodo.6909479.

[2] "LROSE: The Lidar Radar Open Software Environment." http://lrose.net.

[3] "Main Page." http://wiki.lrose.net/index.php/Main_Page.

[4] D. Y. Hancock *et al.*, "Jetstream2: Accelerating Cloud Computing via Jetstream," in *Practice and Experience in Advanced Research Computing (PEARC '21)*, New York, NY, USA: Association for Computing Machinery, 2021, pp. 1–8. doi: 10.1145/3437359.3465565.

[5] T. J. Boerner, S. Deems, T. R. Furlani, S. L. Knuth, and J. Towns, "ACCESS: Advancing Innovation: NSF's Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support," in *In Practice and Experience in Advanced Research Computing (PEARC '23)*, Portland, OR, USA: Association for Computing Machinery, 2023, pp. 173–176. doi: 10.1145/3569951.3597559.

[6] J. Helmus and S. Collis, "The Python ARM Radar Toolkit (Py-ART), a Library for Working with Weather Radar Data in the Python Programming Language," *Journal of Open Research Software*, vol. 4, no. 1, p. e25, 2016, doi: 10.5334/jors.119.

[7] M. Heistermann *et al.*, "The Emergence of Open-Source Software for the Weather Radar Community," *Bulletin of the American Meteorological Society*, vol. 96, no. 1, pp. 117–128, 2015, doi: 10.1175/BAMS-D-13-00240.1.

[8] J. Vivekanandan, D. S. Zrnic, S. M. Ellis, R. Oye, A. V. Ryzhkov, and J. Straka, "Cloud Microphysics Retrieval Using S-Band Dual-Polarization Radar Measurements," *Bulletin of the American Meteorological Society*, vol. 80, no. 3, pp. 381–388, 1999, doi: 10.1175/1520-0477(1999)080<0381:CMRUSB>2.0.CO;2.

[9] T. Matejka and R. C. Srivastava, "An improved version of the extended velocity-azimuth display analysis of single-doppler radar data," *Journal of Atmospheric and Oceanic Technology*, vol. 8, no. 4, pp. 453–466, 1991, doi: 10.1175/1520-0426(1991)008<0453:AIVOTE>2.0.CO;2.

[10] A. Zonca and R. S. Sinkovits, "Deploying Jupyter Notebooks at scale on XSEDE resources for Science Gateways and workshops," in *Proceedings of the Practice and Experience on Advanced Research Computing (PEARC '18)*, New York, NY, USA: Association for Computing Machinery, 2018, pp. 1–7. doi: 10.1145/3219104.3219122.

[11] J. Chastang and A. Espinoza, "Unidata Science Gateway." GitHub, 2017. doi: 10.5065/688s-2w73.

[12] J. Chastang and A. Espinoza, "Advancing Atmospheric Science Education: Customized PyAOS JupyterHubs via the Unidata Science Gateway," in *Proceedings, 40th Conference on Environmental Information Processing Technologies, 104th AMS Annual Meeting*, Baltimore, Maryland, USA, 2024. doi: 10.6084/m9.figshare.25251655.v1.

[13] J. DeHart, T. Barbero, T.-Y. Cha, M. Bell, M. Dixon, and B. Dolan, "LROSE Science Gateway." https://github.com/nsf-lrose/lrose-hub, 2024.

[14] M. M. Bell, M. T. Montgomery, and K. A. Emanuel, "Air-sea Enthalpy and Momentum Exchange at Major Hurricane Wind Speeds Observed during CBLAST," *Journal of the Atmospheric Sciences*, vol. 69, no. 11, pp. 3197–3222, 2012, doi: 10.1175/JAS-D-11-0276.1.

[15] R. M. May *et al.*, "MetPy: A Meteorological Python Library for Data Analysis and Visualization," *Bulletin of the American Meteorological Society*, vol. 103, no. 10, pp. E2273–E2284, 2022, doi: 10.1175/BAMS-D-21-0125.1.

[16] A. Espinoza, J. Chastang, and W. G. Blumberg, "Deploying an Educational JupyterHub for Exploratory Data Analysis, Visualization, and Running Idealized Weather Models on the Jetstream2 Cloud," in *Gateways 2023*, Pittsburgh, PA, USA, 2023. doi: 10.5281/zenodo.10034606.

[17] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007, doi: 10.1109/MCSE.2007.55.

[18] C. R. Harris *et al.*, "Array programming with NumPy," *Nature*, vol. 585, pp. 357–362, Sep. 2020, doi: 10.1038/s41586-020-2649-2.

[19] S. Hoyer and J. Hamman, "Xarray: N-D labeled arrays and datasets in Python," *Journal of Open Research Software*, vol. 5, no. 1, 2017, doi: 10.5334/jors.148.

[20] Met Office, *Cartopy: a cartographic python library with a Matplotlib interface*. Exeter, Devon, 2010. Available: https://scitools.org.uk/cartopy

[21] A. Zonca, "Soft Scaling Kubernetes on Jetstream." https://www.zonca.dev/posts/2024-02-02-soft-scaling-kubernetes-jetstream, Feb. 2024.

[22] T. Lang, B. Dolan, N. Guy, C. Gerlach, and J. Hardin, "CSU-Radarmet/CSU_RadarTools: CSU_RadarTools v1.3." Zenodo, Feb. 2019. doi: 10.5281/zenodo.2562063.

A Fast TIFF File Value Extractor For Generating Timeseries Data to Enable Virtual Climate Stations

Jared McLean
University of Hawai'i - System, Hilo, HI, USA
mcleanj@hawaii.edu

Sean Cleveland
University of Hawai'i - System, Honolulu, HI, USA
seanbc@hawaii.edu

***Abstract*—This paper presents a specialized C++ library that efficiently extracts values from TIFF files to allow for fast generation of timeseries information from a large set of GeoTIFF data. This was produced to assist in generating timeseries for the Hawai'i Climate Data Portal (HCDP), an online portal providing climatological data for the state of Hawai'i. The library utilizes a strip-oriented reading approach and a specialized LZW compression decoder that will process partial strips. A parallelized driver program was created to further enhance extraction efficiency for large sets of data. Experimental results demonstrate a significant speedup for extracting values in early columns from TIFF files. This library is integrated into an API backing the HCDP and is being leveraged to generate on-demand timeseries visualizations for climatological variables at arbitrary points throughout Hawai'i.**

***Index Terms*—TIFF, GeoTIFF, LZW, climate data**

I. Introduction

To improve access to climatological data and accelerate research requiring this data in Hawai'i, the Hawai'i Climate Data Portal (HCDP) [1], [2] was created. The HCDP provides public access to climatological data collected by sensor stations throughout the state, as well as derived data products and visualizations. The primary derived data product provided by the HCDP is a set of gridded data maps representing estimated statewide values at a 250m x 250m resolution. These maps are available for rainfall, temperature, and normalized difference vegetation index, with additional variables in development. These are generated by an automated workflow at a daily and monthly timescale and include 30+ years of historical data.

The Tag Image File Format (TIFF) is a file format commonly used to store raster graphic images – images stored as a matrix of values. The GeoTIFF format is an extension to this that allows for georeferencing data to be stored in the images metadata. This provides a good way to store and display gridded data over a geographical region. GeoTIFFs are used to store climatological data for the HCDP.

Since the HCDP contains a large amount of data over a period of time, it is useful to be able to construct a timeseries representation of the data. While this is relatively straightforward for the sensor station data, which has a relatively limited number of data points which are stored in a database, the ability to view timeseries of "virtual stations" - arbitrary points on the map - is more complicated. This will potentially require processing a very large number of the GeoTIFF files the data is stored in. As such, a method conducive to providing these timeseries to a user without excessive delay requires an efficient method for extracting the values.

This timeseries generator is integrated into an application programming interface (API) [3] that backs the HCDP. The API is set up to index the GeoTIFF data files and can pull a list of files for a given dataset and time period. A TIFF extraction program was created to take this set of files and efficiently generate a timeseries of the data for an index in the gridded data or geographical location. This project is open source and the code can be found at https://github.com/HCDP/geotiff_extract.

II. Background

A number of libraries for handling TIFF and GeoTiff files exist such as tifffile [4] for python and the geotiff.js [5] and tiff.js [6] libraries for JavaScript. These are general purpose libraries and will generally read all of the data in a TIFF file and package it into a class for easy handling. This is great if trying to render the data or manage it in a reusable fashion; however, reading all of the data in a file has a large memory and computational overhead, particularly if the data is compressed.

Lower-level libraries such as libtiff [7] for C++ also exist. Libtiff has the advantage of providing a strip-oriented reading method which allows specific strips to be read. While this is a great improvement over the aforementioned libraries that read and decompress all of the data, with the added advantage of being implemented in a compiled language, it is possible to further speed up this process by decompressing partial strips of data. A further advantage of a specialized library is that it can be very lightweight.

Since decompressing the data composes the majority of the computational overhead of pulling values from a compressed TIFF file, optimizations to this process can have a large benefit to the overall time needed to pull values over a large number of files. To support this ability, a specialized method for extracting an index from the compressed files was developed.

III. Implementation

This TIFF value extractor was implemented in c++ as a header only library. This was designed to support TIFF files using Lempel–Ziv–Welch (LZW) compression, one of the most widely used compression types for TIFF files. The library is broken into two main components, a Reader class that

extracts the necessary data from the file and a Decoder class that handles decoding the LZW compressed data strip.

A. Reader Class

TIFF files are broken into three logical components, the Image File Header (IFH), Image File Directory (IFD), and the image data [8]. The IFH contains some basic information about the file and the file offset for the first IFD. The Reader will read this header to verify the file is valid and then move to the location of the first IFD.

The IFD contains a header indicating the number of entries in the IFD, followed by a set of entries containing tagged values. For the purposes of value extraction, only four values from this section are needed, the map width and height, the offsets for the data strips, and the strip byte counts. The Reader will scan the IFD for these tags and break after getting the required data.

The gridded map data in the TIFF file is stored in strips. Each row of values is considered a strip. These strips may not be stored in a contiguous block of data and have variable lengths when compressed. The strip offset entry in the IFD contains a set of file offsets where each strip is located, while the strip byte counts contains a set of values indicating the size of these strips. This provides all the information needed to read the strip of data containing the requested value. The Reader reads this data strip into a buffer and passes it to the Decoder along with the column of the value being requested.

B. Decoder Class

The LZW decoder is modeled on the decoder used by the python Tifffile library with an important modification. The decoder relies on and builds out a code table for decoding values as it goes. This is built as normal, but the decoded data itself will be ignored until the desired value is reached. Once the value being extracted has been decoded, the Decoder can terminate without decompressing the remaining portion of the data strip. This will result in a variable speed up depending on what column the value being extracted falls in, but the effects can be relatively large particularly for locations close to the left edge of the map, since this decompression is the primary source of computational overhead.

C. Driver

Since the primary purpose for creating this efficient TIFF value extractor is to quickly pull values from a large number of TIFF files, a driver program was also created leveraging this library. The driver takes a set of TIFF files and the requested value index as command line argument, which are provided to the Reader. OpenMP [9] is used to parallelize this process, allowing for a number of files to be processed at once. OpenMP utilizes an internal thread pool to divide the work over an optimal number of threads and is very easy to integrate into the driver program. It uses a pragma compiler directive system allowing the loop handling the calls to the Reader to be parallelized in one line.

Ultimately, this driver code is designed to be integrated into the HCDP API, which will call this as a subprocess and use the results to construct a timeseries of the values. To accommodate this usage, this driver program takes the values and outputs them to stdout in the same order as the files were provided as a space separated list. In the event that any of the TIFF reader functions fail, an underscore is output instead of a value. This output can then be handled by the API and parsed into a JavaScript object notation (JSON) object to be returned to the caller.

Index 0	Index 1143	Index 2287
2.08ms	2.15ms	2.33ms

TABLE I
THE AVERAGE TIME TAKEN FOR THE FIRST, MIDDLE, AND LAST INDEX TO BE EXTRACTED OF A SINGLE FILE.

	Index 0	Index 1143	Index 2287
With Threading	112.83ms	206.36ms	233.64ms
Without Threading	171.41ms	347.21ms	389.36ms

TABLE II
THE AVERAGE TIME TAKEN FOR THE FIRST, MIDDLE, AND LAST INDEX TO BE EXTRACTED OF 400 FILES WITH AND WITHOUT THREADING ENABLED.

IV. Results

The extraction code was tested on the machine running the HCDP API directly. This machine has an Intel(R) Xeon(R) CPU E5-2687W v3 @3.10GHz which has 2 cores. The data files this was tested against have a width of 2288 columns, so the tests were run on column 0, 1143, and 2287 to test the speedup achieved by the partial decompression. At column 2287 the entire strip is decompressed, so this will be analogous to implementations such as the libtiff strip reader that read and decode a single strip. Tests were done on an input of a single file (Table I) as well as on 400 files with multithreading enabled and disabled (Table II). A sample size of 400 was selected since this roughly matches the number of months stored by the data portal's monthly data sets. All tests were averaged over 100 executions.

The tests demonstrated a significant speedup for values at the beginning of a strip vs at the end, with the elements at index 0 taking less than half the time of the last element in the 400 file test, though there does seem to be a more significant speedup at lower indices. Additionally, the threading made a significant improvement in the runtime, with greater improvements being possible if this machine were allocated additional CPUs.

V. HCDP Interface Integration

The API endpoint leveraging this GeoTIFF extraction process was integrated into the HCDP's user interface to generate visualizations of the timeseries of values for climatological variables at any location in Hawai'i. This allows users to select an arbitrary map location and create a "virtual station". The location of this virtual station is sent as a query to the API along with information about the climatological variable being viewed, and a timeseries of data at this location for the lifespan

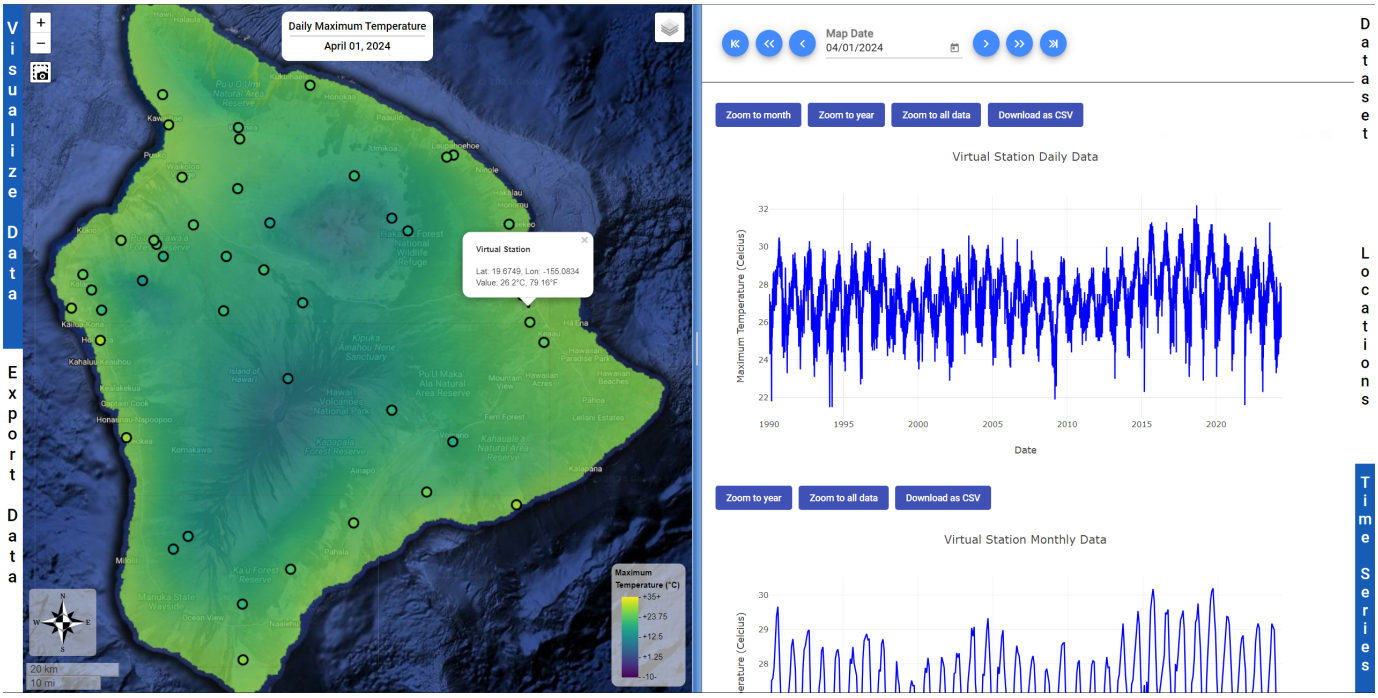


Fig. 1. A timeseries graph of gridded maximum temperature data provided by the HCDP. This data is produced from the gridded GeoTIFF products derived from sensor station data. This data is generated and returned by the GeoTIFF processing endpoint in the HCDP API.

of the dataset is returned. Most of the variables measured are at a daily and monthly timescale. Timeseries for both of these scales are requested and made available to the user.

While the TIFF extraction process described is able to pull values relatively quickly, very large queries can still take some time to process. For example, daily data over the lifespan of a dataset going back to 1990 – the range of many of the HCDP datasets – must index and process over 12,000 files. To improve the responsiveness of the interface and user experience, large queries are separated into multiple chunks. This way data can be loaded into the timeseries visualization as it is received rather than requiring all of the data to be loaded at once. This has the additional benefit of allowing some chunks of a query to be cancelled if a user selects a different location before all of the data from one is complete. The web application throttles the number of requests that are sent to the API at the same time and has a queue of remaining query chunks that can be cancelled before being issued. Single large queries issued to the API whenever a user selects a new location could generate increased load on the API and underlying file system since there is no way to halt an API request once issued.

Once the data is returned to the user it is loaded into a set of interactive graphs for each timescale (Figure 1). Graphs are generated using Plotly.js [10] and have the built-in ability to rescale and manipulate the graph view as well as produce and download an image of the graph. Users can download the timeseries data for the location as a comma separated value (CSV) file containing timestamp and value columns for additional analysis without requiring API access.

VI. Future Work

Currently, this implementation has some limitations. Notably, it only handles files with little endian byte order that use LZW compression or no compression. LZW compression will generally be the most common compression particularly for applications that require a lossless algorithm such as the intended use case for this project; however, it would be useful to extend this to handle other compression algorithms, such as deflate.

Given the intended use case of extracting a value from the same index of GeoTIFF files with the same spatial extent, a further heuristic could be implemented in the case that part of the strips contain background no-data values for a portion of the leading edge of the data. It would be possible to partially decompress the strip from one of the files up to the first real value and begin at that point for the rest of the files, sharing the decompression table state at that position. Assuming all of the files use the same no-data value, the decompression up to this point will be identical; so, this should allow subsequent files to further limit the amount of data they need to decompress.

References

[1] J. McLean, S. B. Cleveland, M. Dodge, M. P. Lucas, R. J. Longman, T. W. Giambelluca, and G. A. Jacobs, "Building a portal for climate data—mapping automation, visualization, and dissemination," *Concurrency and Computation: Practice and Experience*, Nov. 2021. [Online]. Available: https://doi.org/10.1002/cpe.6727

[2] R. J. Longman, M. P. Lucas, J. Mclean, S. B. Cleveland, K. Kodama, A. G. Frazier, K. Kamelamela, A. Schriber, M. Dodge, G. Jacobs, and T. W. Giambelluca, "The hawai'i climate data portal (hcdp)," *Bulletin of the American Meteorological Society*, Apr. 2024. [Online]. Available: http://dx.doi.org/10.1175/BAMS-D-23-0188.1

[3] J. Mclean and S. Cleveland, “Design and implementation of web apis for supporting data product visualization and dissemination in science gateways,” *Proceedings of the 56th Hawaii International Conference on System Sciences*, vol. 10, p. 6966, 2023. [Online]. Available: https://hdl.handle.net/10125/103478
[4] C. Gohlke, “tifffile: Read and write tiff files,” https://pypi.org/project/tifffile/, 2021, accessed: May 12, 2023.
[5] geotiffjs, “geotiff.js: Geotiff reader and writer in javascript,” https://www.npmjs.com/package/geotiff, 2021, accessed: May 12, 2023.
[6] seikichi, “tiff.js: Tiff decoder/encoder in javascript,” https://www.npmjs.com/package/tiff.js, 2015, accessed: May 12, 2023.
[7] S. Leffler and I. Silicon Graphics, *libtiff: Library for Reading and Writing TIFF Files*, 4th ed., LibTIFF Group, San Francisco, CA, August 2021, http://www.simplesystems.org/libtiff/.
[8] International Telecommunication Union, “TIFF specification,” 1992, technical Report. [Online]. Available: https://www.itu.int/itudoc/itu-t/com16/tiff-fx/docs/tiff6.pdf
[9] OpenMP Architecture Review Board, “OpenMP application program interface version 4.5,” 2015. [Online]. Available: https://www.openmp.org/wp-content/uploads/openmp-4.5.pdf
[10] P. T. Inc. (2015) Collaborative data science. Montreal, QC. [Online]. Available: https://plot.ly

User Competence Metrics for Science Gateways: The Case of the KnowCOVID-19 Gateway

MD Ashraful Goni
Media and Communication
Texas Tech University
Lubbock, Texas, USA
mgoni@ttu.edu

Roland Oruche
Computer Science
University of Missouri-Columbia
Columbia, Missouri, USA
ro2q2@mail.missouri.edu

Minhaz Uddin
Media and Communication
Texas Tech University
Lubbock, Texas, USA
minuddin@ttu.edu

Opeyemi Lawal
Media and Communication
Texas Tech University
Lubbock, Texas, USA
oplawal@ttu.edu

Prasad Calyam
Computer Science
University of Missouri-Columbia
Columbia, Missouri, USA
calyamp@missouri.edu

Kerk Kee
Media and Communication
Texas Tech University
Lubbock, Texas, USA
kerk.kee@ttu.edu

***Abstract*—This study provides statistical validation of three composite scales designed to calculate metrics for gateway user competence in terms of domain knowledge, technical skills, and problem-solving orientation. Based on an online survey *(N = 365)* fielded by an online panel company (Centiment.co) with US based participants, analyses using SPSS software demonstrated that technical competence varied between age groups (lower scores for participants aged 60 and higher) and educational levels (lower scores for participants without a bachelor's degree) at a statistically significant level (at 95% confidence interval). These findings suggest that gateway developers may need to provide more technical support to users who are senior researchers and when gateways are being introduced into high school classrooms. Conversely, ethnicity and gender were found to be non-predictors of technical competence. These findings suggest the stereotype of white males being more tech-savvy than other ethnic and gender groups may not hold true anymore.**

***Index Terms*—KnowCOVID-19, User Competence, Competence Metrics, Usability, Science Gateway**

I. Introduction

The COVID-19 disease created both an alarming patient death rate and a data deluge problem for medical professionals during the pandemic. When medical professionals search online about the disease, they drown in a sea of information available on the Internet. However, science gateways can be a solution to this problem. More specifically, our research team developed a (prototype) science gateway augmented by an AI powered chatbot designed to assist medical professionals to search and filter the results based on different levels of evidence [1], so they can focus on a narrower set of literature to identify the information they need to treat their patients. The gateway is called "KnowCOVID-19" and the chatbot is called "Vidura," named after a wise advisor in Indian mythology.

The use of the evidence pyramid [2] is a common approach in the medical profession to filter research papers. For example, a doctor may specifically want to rely on findings from randomized controlled trials (RCTs) and not observational studies. Methodologically, RCTs and observations are different levels of evidence. A doctor can specify which level of evidence to filter their search results on KnowCOVID-19. This approach helps users to more effectively and efficiently find the information they need, and the Vidura chatbot can assist the users on the gateway platform.

However, two issues remain. First, many gateways face limited funding for usability and technical support; funding is mainly for developing open-source prototypes. Many gateways suffer from usability issues, leading to challenges in user adoption and implementation. Second, different users come to the gateway with different levels of competence. Different users may need different answers even if they ask the same question to the AI chatbot. For example, a medical student in training may need more medical explanation about symptoms associated with COVID-19, but a senior medical doctor who is not familiar with online platforms may need more help with technical navigation on the gateway.

Given these two challenges, the present paper seeks to statistically validate our recently developed approach to measure user competence [3]. Measuring user competence can first help gateway developers to identify the users who need more support. Second, two users with different levels of competence (high vs. low in technical competence; high vs. low in COVID-19 knowledge) can be given customized responses based on their competence levels, even if they ask the Vidura chatbot the same question. Having valid user competence metrics is helpful for our gateway and other science gateways across domains. Therefore, we aim to answer the research question, *"How can the user competence metrics (domain, technical, and problem-solving scales) be statistically validated and then be used to generate insights about gateway users?"*

In order to report on the work we set out to accomplish, this paper is outlined as follows. First, we provide a brief review

This project was funded by the US National Science Foundation under grant numbers NSF-2006816 and NSF-2007100. We thank Eric Milman and Chaitra Kulkarni for their assistance in the early usability study.

of the literature on medical information-seeking and user competence. Second, we describe the methods we employed for data collection and statistical validation of our three composite scales for measuring user competence. Third, we present our statistical validations of the scales and how one example of technical competence varied across demographic groups. The findings show how user competence metrics can be used in practical ways to generate user insights. Fourth, we wrap up the paper with a conclusion.

II. Literature Review

A. Medical Information Seeking

Seeking health information about diseases to provide the best treatments to their patients is of great importance to medical professionals [4]. However, how to source health information in a timely fashion and make the best decisions to help their patients was a daunting task for many during the pandemic [5]. When searching for health information, medical professionals often use strategies such as keywords, Boolean Operators, advanced search, and medical synonyms [5]. However, barriers to seeking health information include insufficient time, lack of information search skills, unawareness of accessible sources, high search costs, organizational challenges, location constraints, inadequate information technology infrastructure, and a shortage of medical librarians [4]–[6]. In a systematic review [5] it was reported that medical professionals spend approximately 2 to 32 minutes to finding answers to health questions. In this paper, we believe that what differentiates those who can find the needed information faster than others may be their user competence, especially when it involves a science gateway.

B. User Competence

The notion of competence, originally proposed by White in 1959 [7], [8] as a motivational concept in psychology, has now become a subject of growing research interest across many disciplines. White defined competence as "an organism's capacity to interact effectively with its environment" [8]. In the case of gateways, this can involve an individual's capacity to interact effectively with the technical environment. Similarly, Rychen and Salganik [9] described competence as the individual capacity or capability to effectively fulfill personal or societal requirements, or to perform a specific action or duty. In the case of COVID-19, it can be a medical professional's capacity/capability to effectively find the most rigorous medical information to treat patients. Conversely, many scholars described competence as the observable and measurable attributes of a person, including a mix of their knowledge, skills, abilities, motivations, and self-perception, that lead to outstanding performance [10]. Computer competence encompasses a wide-ranging concept and overlaps with associated terms such as computer experience, expertise, accomplishments, abilities, and literacy. Related to computer competence, Internet competence is conceptualized as a collection of mindsets related to an individual's self-assessed proficiency and comfort with utilizing internet-based tools and platforms [11].

Prior to the present paper, our research team conducted a usability study with 20 participants with KnowCOVID-19 and Vidura chatbot [3]. Participants were assigned various tasks to complete, and their actions were recorded through screen capture videos while they interacted with the gateway. Based on this prior work, we found three types of user competence: medical domain, technical, and problem-solving competence, and we developed three composite scales to measure them in a questionnaire as presented below. Gateway developers can customize the composite scales to fit their own domains.

Medical Domain Competence (User's expertise or specialized knowledge in COVID-19.)

1) When searching for information about COVID-19, I understand the search task at hand.
2) When searching for information about COVID-19, I know the right search terms, keywords, etc., to specify the search.
3) When searching for information about COVID-19, I am able to assess the relevance of search results vs. second-guessing if I have found the answers.
4) When searching for information about COVID-19, I am able to explain the relevance of search results.
5) When searching for information about COVID-19, I am able to tell when the relevant information is found, and the task is done.
6) When searching for information about COVID-19, I can effectively evaluate the credibility of information during my searches.

Technical competence (User's ability to effectively and efficiently use the search engine gateway's features and tools to locate the information they are seeking.)

1) I have experience with basic browser functions (e.g., opening a new tab, sorting, filtering).
2) I have experience with basic keyboard shortcuts (e.g., Ctrl-F, Ctrl-Alt-Delete).
3) I have experience with basic mouse clicks (e.g., right-click for features).
4) I have experience with basic Internet terminologies (e.g., URL, hyperlinks).
5) I can move through the necessary steps for a search task (including browser, keyboard, mouse) logically in sequence vs. missing steps and having to backtrack.
6) I can effectively make use of visual content on web pages and confidently navigate different interfaces on new web pages.

Problem-solving competence (Motivation to adapt themselves to any new innovative technologies to complete the assigned task.)

1) I show some level of calmness and/or enthusiasm when using technology rather than being nervous and/or confused.
2) I show confidence with quick actions when using technology, rather than hesitating or pausing frequently.

3) I am willing to act and try something on a technology even when I am unsure about it.
4) I try another approach immediately when my first attempt does not work while using technology.
5) I am willing to work around usability issues when using a technology.
6) I know when to ask for help and guidance when using technology.

III. Methods and Analysis

For data collection, we contracted with an online survey panel company (Centiment.co) to field the survey with paid participants based in the US between 4/26/2024 and 4/29/2024. Data collection yielded a total of 396 responses. However, upon a close examination, we eliminated 31 responses because these participants failed the attention check embedded in the questionnaire, giving us a final sample of 365 responses for analysis. An attention check (e.g., Please select "All of the above" as the answer for this question) is a fake question in a survey designed to test if a participant picked answers without reading carefully. We assessed participants' level of agreement with the 18 items across three composite scales using a 7-point Likert scale. Prior to collecting survey responses, we obtained IRB approval for the study. Table 1 summarizes the descriptive statistics of key demographic variables of the final sample.

TABLE I
DESCRIPTIVE STATISTICS OF THE DEMOGRAPHIC VARIABLES OF THE FINAL SAMPLE

Demographic Variables	Categories	Number of Participants (Percentage of Sample)
Gender	Females	181 (49.6%)
	Males	183 (50.1%)
	Other	1 (0.3%)
Ethnicity	Whites	254 (69.6%)
	People of Color	111 (30.4%)
	-African Americans	68 (18.6%)
	-Asians/Pacific Islanders	10 (2.7%)
	-American Indians	9 (2.5%)
	-Hispanics	7 (1.9%)
	-Mix	14 (3.8%)
	-Other	3 (0.8%)
Education	Non-Bachelor's Degrees	270 (74%)
	$\geq$ Bachelor's Degrees	153 (26%)
Age Groups	Young Adults (18-29)	64 (17.5%)
	Adults (30-59)	197 (55.0%)
	Older adults ($\geq$ 60)	97 (26.6%)

Based on the final sample *(N = 365)*, we performed a reliability analysis using the SPSS (Statistical Package of Social Sciences) Software. Specifically, a reliability analysis refers to the process of measuring the consistency of the items in a composite scale based on inter-item correlations. In other words, let's take the medical domain competence scale with six items discussed earlier as an example: if the six items in the composite scale share a high level of inter-item correlations above 0.70, then the composite scale is deemed consistent enough to converge as a coherent measure of the concept of medical domain competence. Similarly for the composite scales of technical and problem-solving competence.

This inter-item correlation score is called Cronbach's alpha, where the value of 1 means a perfect 100% correlation among all 6 items, and a value of 0 means no correlation at all. With a satisfactory alpha score, a composite score for each type of competence can be calculated by averaging the six individual item scores within the three respective composite scales. Then the three composite scores will serve as the domain, technical, and problem-solving competence metrics.

IV. Findings

Based on our analysis, the medical domain competence scale achieved a satisfactory alpha score (α = .93), similarly for the technical competence scale (α = .93) and the problem-solving competence scale (α = .89). While 0.70 is commonly considered the minimum score for a reliability analysis, some sources suggest that a 0.60 score may be acceptable, especially for a scale in progress. However, statistically, our three scales achieved a high degree of reliability.

Recall that we discussed averaging the individual item scores to obtain three composite scores (metrics). These metrics can be used to assess and evaluate the three types of competence in the case of using the KnowCOVID-19 gateway. However, social scientists can use the metrics to explore their relationships with other variables in the same questionnaire. We explored how the three scores varied across different groups based on gender, ethnicity, age, and education at the 95% confidence level ($p < 0.05$). Due to space limitations, we only report findings of technical competence as a case in point.

1) Ethnicity: An independent sample *t*-test assessed whether technical scores differed between Whites and people of color. White participants (M = 5.72, SD = 1.18) did not differ significantly from people of color (M = 5.98, SD = 1.23), $t(363)$ = -1.89, p = .059. Because the difference is not statistically significant, this finding suggests that ethnicity is not a generalizable predictor of technical competence, although people of color scored higher than Whites in the sample.

2) Gender: An independent sample *t*-test was conducted to compare the technical competence between males and females. There was no significant difference between males (M = 5.82, SD = 1.23) and females (M = 5.77, SD = 1.17; $t(362)$ = 0.35, p = .730). This result suggests that gender is not a statistically significant predictor of technical competence.

3) Age Groups: A one-way between-groups *ANOVA* was conducted to assess the differences in technical competence across different age groups: young adults (18-29), adults (30-59), and older adults ($\geq$ 60). The test results showed significant differences in technical competence ($F(2, 355)$ = 15.63, $p < .001$, $\eta^2 = .081$) across age groups. Post hoc comparisons using Tukey's HSD test further showed that adults (M = 6.05, SD = 1.06) had significantly higher technical competence compared to older adults (M = 5.26, SD = 1.22, $p < .001$). Additionally, young adults (M = 5.91, SD = 1.28) also had significantly higher technical competence compared to older adults ($p < .001$). However, adults did not differ

significantly from young adults in terms of technical scores (p = .694). Therefore, the significant differences lie primarily between older adults and the other two groups. The partial eta squared value of .081 indicates a moderate effect size.

4) Education: An independent sample t-test was conducted to compare technical competence scores for participants based on education attainment at two levels (bachelor's degree or higher vs. non-bachelor's degree holders). There was a significant difference in scores between participants with at least a bachelor's degree (M = 6.05, SD = 1.05) and non-bachelor's degree holders (M = 5.71, SD = 1.23). The t-test results indicated a statistically significant difference, t (363) = -2.39, p = .018. The result suggests having a bachelor's degree or higher is a predictor of technical competence.

V. Conclusion and Future Research

In conclusion, with a sample of 365 participants, we statistically validated three previously developed composite scales to measure user competence in terms of medical domain knowledge, technical ability, and problem-solving orientation. The data collection involved an online survey fielded by Centiment.co, and the reliability analysis was performed using SPSS software to ensure the internal consistency of these scales. The three composite scales achieved satisfactory Cronbach's alpha scores, confirming their reliability for future use as user competence metrics. These validated metrics were then employed to assess competence levels across demographic groups, providing valuable insights into variations in user competence.

Furthermore, we demonstrated how technical competence scores (as an example for demonstration in this paper) varied across age groups and educational levels. Specifically, adults (ages 30-59) and young adults (ages 18-29) both scored higher than older adults (aged 60 and above) at a statistically significant level. This finding suggests that age 60 is the demarcation point, where users aged 59 or younger are technically more competent. Gateway developers may need to provide more onboarding and technical support for users aged 60 or older.

Additionally, participants with at least a bachelor's degree also scored higher than participants without a bachelor's degree in terms of technical competence. Because the majority of gateways are being used by graduate students and faculty who hold a bachelor's degree or higher, gateway developers could expect a certain level of technical competence. Gateway developers may need to provide more technical support to users when a gateway is being introduced into a high school classroom. Future research should investigate the technical competence of undergraduate students who are working towards their bachelor's degrees.

Conversely, ethnicity and gender were not found to be predictors of technical competence at a statistically significant level. Gateway developers may be able to take comfort in our results, which suggest that the digital divides between ethnic and gender groups may have been successfully bridged, at least in our study sample. In other words, the stereotype of white males being more tech-savvy than other ethnic and/or gender groups may not be true anymore today, at least in the case of using online technologies to search for COVID-19 information.

How else can the user competence metrics be used to support the adoption of science gateways? We measured participants' degree of agreement with the 18 items across three composite scales using a 7-point Likert scale. Given this, the value of 4-point represents the midpoint of the Likert scale. Generally, one can consider an individual composite score of 3.9 or below to be "low" and a score of 4.0 or higher to be "high". Other demarcation variations (e.g., using the mean or median instead of the midpoint) can be the judgments of the gateway developers within their particular contexts. Let's say we use 4 as the midpoint of the scale and divide all the users of a gateway into two groups of high vs. low domain knowledge (specific gateway domain), technical, and problem-solving competence, then we can create a 2x2x2 matrix of eight different quadrants. This means each gateway user can be placed in one of these quadrants (e.g., high in domain, low in technical, and high in problem-solving), thus allowing the AI chatbot to customize the responses as discussed in the introduction. However, users are likely to improve on the three competence metrics over time, thus being able to transition from one quadrant to another. This approach can allow our AI chatbot to further customize its responses to the users based on their latest position in the matrix.

References

[1] B. W. Shibo, R. Oruche, M. A. Goni, X. Cheng, E. Milman, P. Calyam, and K. Kee, "Challenges faced by medical professionals as gateway users: The case of knowcovid-19," in *Proceedings of Science Gateways 2023 (SG23)*, Pittsburgh, PA, October 30 2023.

[2] M. H. Murad, N. Asi, M. Alsawas, and F. Alahdab, "New evidence pyramid," *BMJ Evid.-Based Med.*, vol. 21, no. 4, pp. 125–127, 2016.

[3] M. A. Goni, R. Oruche, M. Uddin, O. Lawal, P. Calyam, and K. E. Kee, "User competence metrics for cyberinfrastructure: The case of know covid-19 science gateway," in *Metrics 2023*, Denver, Colorado, November 2023.

[4] M. Andualem, G. Kedebe, and A. Kumie, "Information needs and seeking behavior among health professionals working at public hospitals and health centers in bahur dar, Ethiopia," *BMC Health Services Research*, vol. 13, p. 534, 2013. [Online]. Available: https://www.biomedcentral.com/1472-6963/131534

[5] A. Daei, M. R. Soleymani, H. Ashrafi-Rizi, A. Zargham-Boroujeni, and R. Kelishadi, "Clinical information seeking behavior of physicians: A systematic review," *International journal of medical informatics*, vol. 139, p. 104144, 2020.

[6] F. A. Geda, "The roles of medical library in information seeking behavior of health care professionals: A review of literature," *Journal of Hospital Librarianship*, vol. 21, no. 4, pp. 405–416, 2021.

[7] K. Schneider *et al.*, "What does competence mean?" *Psychology*, vol. 10, no. 14, p. 1938, 2019.

[8] R. W. White, "Motivation reconsidered: the concept of competence," *Psychological Review*, vol. 66, no. 5, p. 297, 1959.

[9] D. S. Rychen and L. H. Salganik, "Definition and selection of competencies (deseco): Theoretical and conceptual foundations. strategy paper," 2002, neuchatel, Switzerland: Swiss Federal Statistical Office.

[10] L. M. Spencer and S. M. Spencer, *Competence at Work: Models for Superior Performance*. John Wiley & Sons, 2008.

[11] B. W. Wirtz, R. Piehler, and P. Daiser, "E-government portal characteristics and individual appeal: An examination of e-government and citizen acceptance in the context of local administration portals," *Journal of Nonprofit & Public Sector Marketing*, vol. 27, no. 1, pp. 70–98, 2015.

Locking Down Science Gateways

Steven R Brandt*, Patrick Diehl‡*†
*LSU Center for Computation & Technology, Louisiana State University, Baton Rouge, LA, 70803 U.S.A.
† Department of Physics and Astronomy, Louisiana State University, Baton Rouge, LA, 70803 U.S.A.
‡ Applied Computer Science (CCS-7), Los Alamos National Laboratory, Los Alamos, NM 87545 U.S.A.

***Abstract*—The most recent Linux kernels have a new feature for securing applications: Landlock. Like Seccomp before it, Landlock makes it possible for a running process to give up access to resources. For applications running as Science Gateways, we want to have network access while starting up MPI, but we want to take away network access prior to the reading of parameter files in order to prevent malicious exploits of the gateway code. We explore the usefulness of this tool by modifying and locking down two mature scientific codes: The Einstein Toolkit, and Octo-Tiger.**

***Index Terms*—Science Gateways, security, landlock**

I. Introduction

Science Gateways typically provide a graphical or web interface to scientific code, allowing users who are less savvy about the command line, supercomputers, Slurm, etc. to have ready access to advanced codes. Often, in the interest of democratization, the vetting process for users of the Gateway is less rigorous than a typical user account. Because scientific codes are typically written in C, C++, or Fortran without a thought about security, these applications represent a potential security hazard via buffer overruns, poor input sanitization, etc. Full audits of these codes represent a cost few are willing to undertake.

An ideal solution for these systems would be to sandbox the code, limiting what it can do even if a hacker were to gain control of the running process. Because these applications typically run in a distributed fashion over MPI, they need the ability to turn on the sandbox (and take away the ability to make new connections) after calling `MPI_Init()`. In addition, of course, the sandbox should limit what directories the process can read or write.

The most recent Linux kernel (version 6.9), fortunately, offers a way to lock down an application, *i.e.* a way for an application to give up its access to the network and to files. Can this tool (landlock) readily address the needs of the Science Gateway and make sophisticated scientific applications secure? Can it accomplish this without overburdening the Gateway developer with the need to understand deep things about security or make extensive modifications of their code?

In this paper, we explore the difficulty in securing two codes: The Einstein Toolkit, which we will use to simulate a spherically symmetric neutron star, and Octo-Tiger, which we will use to simulate a white dwarf. These codes have little in common except that they are large C++ codes that explore astrophysical scenarios. We will show that it is relatively easy to modify these codes to employ Landlock.

The paper is structured as follows: Section II discusses security tools and methodologies. Section III briefly introduces the studied scientific applications. Section IV addresses the implementation and testing. Section V shows examples to lock the file systems access and network. Section VI shows some run time measurements for Octo-Tiger with and without `Landlock`. Finally, Section VII concludes the work.

II. Security Tools and Methodologies

Landlock is far from the first tool designed to lock down an otherwise insecure application and prevent it from doing malice. The original Linux had `chroot` to serve this purpose. The `chroot` system call changed the effective root of the process calling it. It thereby gave up access to all files below the new root passed to it. Unfortunately, subsequent calls to `chroot` can *undo* the first call, so it is inadequate even for a low-level sandboxing. The `pivot_root` system call, introduced in Linux 2.3.41, provides an irreversible change of the root directory. While it does make it possible to sandbox direct access to the file system from the current process, it does not prevent the local process from opening network connections, creating IPC resources, etc. So it is, at best, a start at building a sandbox.

Another, more comprehensive effort at limiting what applications can do, is provided by the `seccomp` facility and has been available since Linux 3.17. This tool allows system calls to be selectively blocked and filtered. However, `seccomp` does not claim to be able to sandbox an application, but it does provide a way for an application to give up a wide variety of privileges. One limitation of `seccomp` is that one cannot pass pointers to it. This means it cannot be given character arrays, and this means it cannot be used to limit access to specific files or directories. In principle, it could be combined with `pivot_root` to accomplish this end.

`SELinux` and `AppArmor` provide true sandboxing capabilities, but they must be configured by the root user and provide system-wide restrictions. It should, in theory, be possible to configure them into an image and launch the image from MPI. In principle, these tools could be configured for an Apptainer image and launched by a user. Alternatively, rules could be crafted collaboratively between the sysadmins and the Science Gateway developer. While these tools are fully capable of providing the necessary level of restriction, turning them on after calling `MPI_init()` might, however, prove challenging.

OpenBSD has similiar capabilities to `Landlock` through its `pledge` and `unveil` system calls. However, very few

clusters currently use OpenBSD on their clusters. Namespaces also have the capability of limiting what an app can do, but they were designed more for virtualization than for security. `Landlock`, however, allows programmers the flexibility of controlling when restrictions are turned on and requires neither special permission from sysadmins nor virtualization. We believe that, in many cases, this will make it the best choice for locking down a science gateway.

III. Scientific applications

Although theoretically present in kernel version 5.13, the first version in which `Landlock` was capable of stopping network connections seems to be Fedora 40 running kernel version 6.8.1. We feel that this is a crucial capability for the purposes for preventing bad actors from gaining control of or misusing local resources.

We constructed a function call named `landlockme()` [1] which our applications can call. It is based on an example landlock sandboxing code found here [2]. This code uses environment variables to communicate which directories the application is allowed to read, write, and where (if anywhere) it is allowed to make internet connections and on what ports.

Most scientific codes follow a standard workflow: (1) initialize MPI, (2) then read parameter and/or data files, (3) and then finally produce a result. To secure such an application, one inserts a call to `landlockme()` or the equivalent between steps (1) and (2). For this strategy to be effective, the gateway should not give the user any control over command line arguments to the application, only to the contents of the parameter and input files.

The insertion of this call can be performed in one of two ways: (1) editing the source code, or (2) using PMPI to call `MPI_Init()`.

A. The Einstein Toolkit

The Einstein Toolkit (ET) [3] is a hybrid code constructed from C, C++, and Fortran. It's core infrastructure was first created in 1977 and it has been under continuous development since. While the core infrastructure, Cactus, is generic and could be used for any Cauchy problem, the family of science-specific modules in the ET centers on fully relativistic astrophysical simulations, *e.g.* black holes, neutron stars, supernovae, and cosmology. Cactus provides adaptive mesh refinement (AMR) with subcycling in time. Using the Carpet driver, this is in the form of nested and moving boxes rather than a fully general refinement system.

The test problem we are using in this paper is a TOV star [**?**]. This is a simple, spherically symmetric neutron star which we model on a full 3D Cartesian grid. While this is more computational infrastructure than is needed for such a simple simulation, it is a common test problem that is run to verify code correctness and to teach students about neutron stars and the ET code. The TOV star will exercise all important components of the solvers required for more sophisticated problems.

B. Octo-Tiger

Octo-Tiger is an astrophysical code simulating the evolution of non-relativistic star systems using adaptive octrees [4]. Octo-Tiger simulates the following multi-physics: Gravity is solved using a fast-multipole method (FMM) and the hydro equation is solved using a finite volume method with a fully adaptive mesh refinement (AMR) without subcycling in time (subcycling is avoided because of the need to solve elliptic equations). Octo-Tiger is implemented in C++ using the C++ standard library for parallelism and concurrency (HPX) [5].

IV. Implementation and Testing

We note that while `Landlock` works on Fedora 40, we were unable to get the default installed `valgrind` to work. As far as `valgrind` and the emulated CPU it uses work, the system does not have the capability. The GNU debugger project (gdb) worked with `Landlock`.

A. The Einstein Toolkit

We began by testing very basic MPI codes that exchange simple messages of random data using `MPI_Send` and `MPI_Recv` in order to verify whether our method works.

We were able to show that if `landlockme()` was called before `MPI_Init`, then the application did not run. If we called `landlockme()` after `MPI_Init`, then the application ran without difficulty using MPICH. When we attempted the same test using OpenMPI, `Landlock` blocked an attempt to use shared memory. In principle, we could ask OpenMPI not to do this, or we could change the rules to allow shared memory. For simplicity, we tested our scientific codes using MPICH.

The modification to the Einstein Toolkit was straightforward. We were able to identify the function call `CCTKi_InitialiseCactus` and insert the call to `landlockme()` after the call to a method named `CCTKi_InitialiseDataStructures`. With the addition of this single line of code, our TOV star example was able to run and generate data files.

As a double check that the `Landlock` was indeed active, we ran the tests again, running it in directories it was not supposed to be able to access. As expected, `Landlock` prevented it from reading or writing files.

B. Octo-Tiger

For the Octo-Tiger version without networking, we called `landlockme()` as the first thing after entering the main method. As expected, `Landlock` prevented it from reading or writing files. For the version with networking on, we had to make sure that each MPI rank calls `landlockme()`. Here, we called the function in the initialization of each MPI rank. For debugging purposes, we added simple stdout/stderr messages to the `landlockme()` function. So the function is called for each MPI rank. Adding `LandLock` to Octo-Tiger was straightforward and had no major issues. We have to mention that we had to do code changes unrelated to `LandLock` but specific to GCC 14. We are in the process to prepare a pull request for adding `LandLock` as an optional feature to Octo-Tiger.

Listing 1. Example variables to land lock applications.

```
export LL_FS_RO="/bin:/lib/:$USER/"
export LL_FS_RW="$USER/"
export LL_TCP_BIND=""
export LL_TCP_CONNECT=""
```

C. Generic Science Codes

We note that, with `Landlock`, it is theoretically possible to create a service which runs arbitrary MPI codes on behalf of unknown users. One way to accomplish this would be to accept a user code in the form of a shared library (*i.e.* a `.so` file) with some kind of standard method, *e.g.* `runcode()`. The service would call `MPI_Init`, then `landlockme()`, then it would use `dlopen()` and `dlsym()` to access and run the user method. By using `dlopen()` instead of linking the shared object file, we circumvent the potential problem of constructors being called prior to `landlockme()` in C++'s initialization sequence.

V. Configuration

Listing 1 shows some of the configuration options we used in this study. The first option is `LL_FS_RO`, which takes a list of paths separated by colons. These are the files and directories the application is allowed to read. We gave access to the system-wide installed libraries and executables. The second option `LL_FS_RW` provides a list of files and directories the application is allowed to write to. The third option `LL_TCP_BIND` restricts the port binding and the fourth options `LL_TCP_CONNECT` restricts the ports for connections. We refer to the Linux kernel documentation [6] for more options.

VI. Runtime measurements

Using `landlockme()` should not introduce any overhead to the process. We executed the rotating star problem from Octo-Tiger's test suit to investigate the claim. We adaptive refined the intial mesh four times and executed the simulation for ten steps. Both runs with and without landlock took around 92 seconds. We compiled Octo-Tiger using Spack [7] and restricted it to access the Spack installation directory and the user's home directory to read and write the output files. We could not observe introduced overheads for the Octo-Tiger version with networking.

VII. Conclusion

In this work we have studied the use of `Landlock` for securing scientific applications for use in Science Gateways. Because Science Gateways typically involve taking large, mature, C/C++ and Fortran codes and making them semi-publicaly available on the web, they represent a potential security hazard. These codes usually are based on MPI and follow a pattern of starting up, reading initial data files, computing, then generating results. Modifying such codes to invoke `Landlock` after MPI startup but before the reading of parameter files should secure the code against attackers launching attacks based on input files (e.g. exploiting buffer overruns or unsanitized inputs).

We note, however, that `Landlock` provides no protection against denial of service attacks, *e.g.* using up file space, inodes, file descriptors, etc. While these types of threats can still cause significant problems, they are of a different class. They should not allow user data or password information to be stolen, back doors to be installed, etc.

We have demonstrated that, even with large, complex codes such as Octo-Tiger and the Einstein Toolkit, sandboxing a code with `Landlock` is a relatively straightforward task. Not only was this done with relative ease, it introduced no performance penalties or observable runtime overheads.

Appendix A
Supplementary materials

Octo-Tiger is available on GitHub [8] and can be compiled with Spack [9]. The scripts and input data to reproduce the runs in Section VI are available on Zenodo [10].

The Einstein Toolkit is free under the GPLv3 license and is available for public download using instructions found at the Einstein Toolkit website [11].

Acknowledgment

The authors would like to thank the IT support staff of the Center for Computation and Technology who setup our test machines for us. Also, we wish to acknowledge the support of NSF grant OAC 2004157 to support work on the Einstein Toolkit. This work was supported by the U.S. Department of Energy through the Los Alamos National Laboratory. Los Alamos National Laboratory is operated by Triad National Security, LLC, for the National Nuclear Security Administration of U.S. Department of Energy (Contract No. 89233218CNA000001). LA-UR-24-27511

References

[1] "landlock(7) — linux manual page," last accessed 07/20/2024. [Online]. Available: https://man7.org/linux/man-pages/man7/landlock.7.html

[2] S. R. Brandt, "Sandbox for landlock," last accessed 07/18/2024. [Online]. Available: https://gist.github.com/stevenrbrandt/ced0bd99a90628453cbd899480d435d2

[3] "The einstein toolkit," 2023. [Online]. Available: https://doi.org/10.5281/zenodo.10380404

[4] D. C. Marcello, S. Shiber, O. De Marco, J. Frank, G. C. Clayton, P. M. Motl, P. Diehl, and H. Kaiser, "Octo-tiger: a new, 3d hydrodynamic code for stellar mergers that uses hpx parallelization," *Monthly Notices of the Royal Astronomical Society*, vol. 504, no. 4, pp. 5345–5382, 2021.

[5] H. Kaiser, P. Diehl, A. S. Lemoine, B. A. Lelbach, P. Amini, A. Berge, J. Biddiscombe, S. R. Brandt, N. Gupta, T. Heller *et al.*, "HPX-the C++ standard library for parallelism and concurrency," *Journal of Open Source Software*, vol. 5, no. 53, p. 2352, 2020.

[6] M. Salaün, "Landlock: unprivileged access control," 2024, last accessed 07/17/2024. [Online]. Available: https://docs.kernel.org/userspace-api/landlock.html

[7] T. Gamblin, M. LeGendre, M. R. Collette, G. L. Lee, A. Moody, B. R. de Supinski, and S. Futral, "The Spack Package Manager: Bringing Order to HPC Software Chaos," ser. Supercomputing 2015 (SC'15), Austin, Texas, USA, November 15-20 2015, lLNL-CONF-669890. [Online]. Available: https://github.com/spack/spack

[8] D. Marcello *et al.*, "Octo-tiger: Astrophysics program simulating the evolution of star systems based on the fast multipole method on adaptive octrees," last accessed 07/15/2024. [Online]. Available: https://github.com/STEllAR-GROUP/octotiger

[9] G. Daiß, J. Yan, P. diehl, and C. Junghans, last accessed 07/18/2024. [Online]. Available: https://github.com/G-071/octotiger-spack
[10] P. Diehl, "Data: Locking down science gateways," May 2024. [Online]. Available: https://doi.org/10.5281/zenodo.11355929
[11] "The einstein toolkit," last accessed 07/15/2024. [Online]. Available: https://einsteintoolkit.org

Designing and Deploying a FAIR Resource Portal for Geospatial Data

Yiqing Qu
Computer and Information Technology
Purdue University
West Lafayette, USA
qu112@purdue.edu

Christopher Thompson
Rosen Center for Advanced Computing
Purdue University
West Lafayette, USA
thompscs@purdue.edu

Rajesh Kalyanam
Rosen Center for Advanced Computing
Purdue University
West Lafayette, USA
rkalyana@purdue.edu

***Abstract*—In recent years, the need for FAIR (Findable, Accessible, Interoperable, and Reusable) data portals has become increasingly important for effective utilization of the wealth of data that is being generated through growing access to computational resources. Despite this growing demand, there are currently no canonical open-source solutions available for FAIR-compliant resource data portals. This paper reports on our preliminary results in trying to address this critical gap by providing a practical implementation strategy for deploying FAIR-compliant portals. Specifically, this paper presents our experience in designing and systematically implementing a FAIR-compliant resource portal for geospatial data. The paper outlines the system architecture, including the high-level design, data resource management, and metadata extraction processes.**

***Index Terms*—FAIR,portal,gateway,geospatial**

I. Introduction

The FAIR data principles proposed in 2016 [1] have gained broad acceptance and importance in data management strategies for research projects. At the same time, a variety of FAIR evaluation tools have been designed [2] that can help evaluate the FAIR compliance of a particular data portal. However, gateway (and more specifically, data portal) developers still lack clear guidelines on how to implement these principles in practice. Existing solutions for data portals often fall short in fully adhering to FAIR principles, particularly in terms of interoperability and reusability.

Our project aims to address these gaps by developing a FAIR-compliant resource portal specifically designed for geospatial data[1]. While our focus on geospatial data is in response to the needs of a resource management portal for managing workflow outputs from the GeoEDF workflow engine [3], we believe that our portal design and implementation strategies can be adapted for any other scientific domain.

The development of this portal involved several key steps, including the design of a robust system architecture around an extensible open source portal framework, the integration of domain-specific metadata extraction and standardization, and the implementation of features that ensure adherence to FAIR principles. We describe each of these steps in the following sections and conclude with a summary of the lessons learned during our implementation.

[1]https://geoedf-portal.anvilcloud.rcac.purdue.edu/

II. Background

A. GeoEDF

The Extensible Geospatial Data Framework (GeoEDF) [3] is designed to reduce the amount of effort geospatial researchers spend in wrangling data for their research workflows. GeoEDF provides a plug-and-play workflow engine that enables researchers to integrate data acquisition and processing operations into their workflows through community-contributed, reusable workflow building blocks termed "data connectors" and "data processors" respectively. GeoEDF also seeks to support the FAIR principles by enabling researchers to publish both their workflows and the workflow outputs to a data portal where other researchers can discover and reproduce these workflows. Rather than utilize an existing geospatial data portal such as Hydroshare [4], the GeoEDF team sought to develop a FAIR-compliant portal from the ground up in order to systematically identify the components of FAIR compliance and the means to address each of these requirements. The portal described in this paper is the result of this endeavor.

B. Metadata Schema

A key component of FAIR compliance is the use of a structured metadata schema to describe the hosted datasets. Common schemas used in describing data include Schema.Org, ISO 19115 and Dublin Core. These schemas provide standardized elements for documenting various aspects of data. We selected Schema.org for our portal due to its rich semantics and flexibility for geospatial data [5]. ISO 19115 while highly formalized and comprehensive is not specifically designed for web applications. Dublin Core although more web-friendly, lacks specific geospatial properties and hierarchical structures that are useful for describing complex resources [6]. In contrast, Schema.org supports detailed properties like spatial coverage, making it suitable for comprehensive geospatial data descriptions. Its ability to embed elements allows us to represent a resource with multiple files as a single, coherent entity. Additionally, Schema.org's adoption by Google enhances the discoverability of datasets. Its established framework ensures consistency and interoperability [7].

C. Globus Modern Research Data Portal

The first step in designing the data portal was identifying a suitable extensible open-source portal framework that could serve as the baseline for further customization. We chose the Globus Modern Research Data Portal [8] both due to its extensibility as well as its integration with Globus authentication, data management, and search. Globus Auth, the authentication system, integrates with institutional single sign-on services for secure access and privacy compliance. Globus Search enhances data discoverability with advanced indexing and search capabilities based on metadata, content, and queries. Furthermore, since the portal is based on the Django framework, this readily provides a collection of tools that can be used to implement other features such as a data publication API.

III. Portal Design

The system architecture of our data portal is shown in Figure 1 and illustrates the custom-developed components as well as existing services and tools that have been leveraged in our implementation. At its core is a customized Globus Django Data Portal that serves as the central hub, managing a set of searchable published data resources through integration with a Resource Database and a dedicated Globus Search Index. Each published data resource has a dedicated resource landing page, which displays detailed metadata (including where available, spatial coverage) and includes embedded Schema.org metadata for indexing via web crawlers, leading to enhanced web searchability. Resource publication is primarily via a publication API that can be invoked from a variety of client interfaces such as an external JupyterHub or a built-in file manager. On invocation, the publication API triggers asynchronous metadata extraction and indexing to Globus via the RabbitMQ message broker and a scalable set of Resource Metadata Extractor workers. The FAIRness Evaluator assesses published resources for compliance with the FAIR principles. Finally, the portal integrates with Google, allowing it to crawl and index resources, thus displaying rich results in search queries.

A. Resource Management

We define a resource as any dataset or file collection that users wish to publish and share. We categorize resources into three types to streamline their management and metadata extraction:

- Geospatial Files: This type includes vector and raster files with geospatial data. For these files, we extract metadata such as geospatial coverage and projection information which can be used in search and display.
- Workflow: This category includes resources related to GeoEDF workflows, which is a design specific to the GeoEDF project. In order to enable reproducibility, these resources include workflow input files (if any), the workflow definition file in the YAML format, and the workflow output files. The resulting resource is packaged as a single zip file, with metadata being extracted for each of the components.
- General Files: This type includes all other datasets that do not fall into the previous categories. These files can include various types of research data, and we extract general metadata such as file type, size, and descriptive information provided by the user.

The Resource Database manages administrative information about each resource such as the uniquely assigned resource ID, the user requesting its publication, and its publication status as it goes through the process of metadata extraction and registration in the Globus Search Index.

B. Metadata Extraction

The metadata extraction process is designed to extract metadata from various file types and submit it to Globus, ensuring resources are well-described and discoverable. A resource publication request via the publication API results in a message routed to the RabbitMQ broker. Using RabbitMQ for asynchronous communication prevents the loss of publication requests, which might occur due to the long processing time for large files or errors from external services. Asynchronous processing enhances fault tolerance and reliability, ensuring that the system can handle long-running tasks and recover from failures without losing data.

A scalable set of metadata extraction workers continuously listen for new messages on the configured RabbitMQ queue. Upon receiving a message, the corresponding resource files are processed to extract and assemble metadata in the Schema.org format and submitted to Globus Search via the use of the Globus SDK. Periodic polling is utilized to enquire the status of the metadata ingestion into Globus Search, which is then updated in the Resource Database.

C. Making the Portal FAIR

As a first step to ensuring the FAIRness of the data portal, we sought to ensure that each "findable" resource (that has a unique resource identifier) has sufficient discoverable metadata in the resource landing page reachable via its unique identifier. Additionally, we also sought to implement a rich resource API that will support programmatic metadata discovery as well as integration with external tools.

1) Embedded Schema.org metadata: Resource metadata is compiled into a Schema.org JSON by mapping the extracted metadata to corresponding Schema.org properties. The resulting Schema.org JSON-LD metadata is then embedded directly into the HTML source of the landing page of each resource using Django derived templates. To ensure that the embedded metadata is recognized and validated by Google, our implementation follows the documentation and guidelines provided by Google Dataset Search [9]. The key elements included in the Schema.org object are basic information such as name, size, and modification time; unique identifiers for findability; a detailed description providing contextual information; information about the resource creator; geospatial and

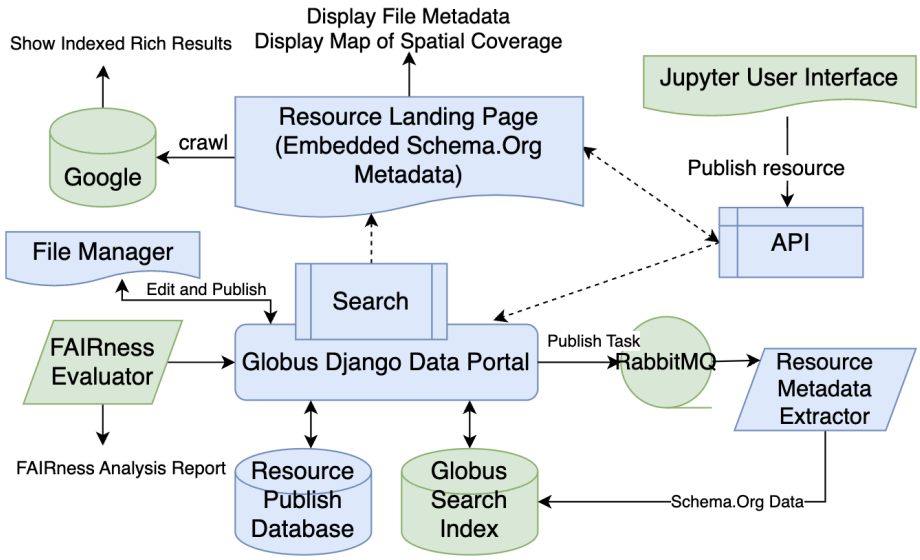


Fig. 1. System Diagram illustrating the custom-developed components (blue) and leveraged services and tools (green).

temporal coverage details; URLs for accessing the resource and downloading it; and licensing details promoting reuse.

2) Resource API: The Resource API complements the data portal and is designed to support programmatic management and discovery of the published resources and their metadata. Token-based authentication is used for access control and to verify user identity before performing certain data operations. The API supports various operations, including publishing resources and retrieving metadata. Parameters are logically arranged for easy parsing and processing to ensure clear user interaction with the API[2].

IV. Portal Deployment

In order to ensure scalability and portability, the portal and associated components (such as the metadata extractor) are deployed using Docker containers and Kubernetes orchestration in Purdue's Anvil composable cloud. Docker image builds for the portal and metadata extractor are automated through GitHub Actions to streamline the CI/CD process and push container images to the Anvil composable cloud's Harbor registry. The various deployment files and configurations are available in our public GitHub repositories [10, 11].

V. FAIRness Evaluation

To further ensure the FAIRness of the data portal, we next applied a data-driven, systematic approach to improving its quantitative score using well-known FAIRness evaluators.

A. Evaluator Selection

In assessing various FAIRness evaluators we tested three tools: F-UJI [12], FAIRsharing [13], and FAIRshake [14]. While each evaluator had its strengths, F-UJI proved to be the most mature in automatically assessing aspects of FAIR compliance with well-developed assessment metrics as well as providing detailed feedback on missing elements that impact the assessment score.

B. Evaluation Result

The evaluation results obtained from F-UJI are presented in a structured JSON format, which includes metric identifiers, names, outputs, statuses, and scores for a total of 16 FAIR evaluation metrics. In order to perform a systematic improvement of our data portal, we first carried out a preliminary evaluation following our initial implementation. This evaluation revealed gaps in persistent identifiers, comprehensive metadata, and integration with semantic resources, resulting in a FAIR compliance score of 47% (Figure 2). After implementing targeted improvements to address these specific areas, the portal's score increased to 60% (Figure 3), reflecting the positive impact of these enhancements on FAIR compliance.

Broadly, there are certain key considerations in ensuring FAIRness as revealed by the evaluation. In terms of findability, our portal effectively uses globally unique identifiers and supports metadata retrievability. However, there is room for improvement in the use of persistent identifiers and comprehensive descriptive metadata. For accessibility, the portal show full compliance in providing secure data and metadata access through network protocols, ensuring clear and reliable data access conditions. Regarding interoperability, the use of JSON-LD for metadata structuring of linked and composite resources is effective. When it comes to reusability, our portal satisfies requirements in data content specification and clear licensing for reuse.

VI. Discussion and Future Enhancements

Based on the evaluation, we identified several areas of future improvement to enhance the FAIR compliance of our portal.

[2]https://geoedf-portal.anvilcloud.rcac.purdue.edu/swagger

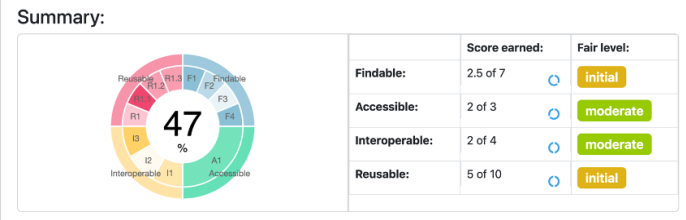


Fig. 2. FAIR evaluation report on the initial version

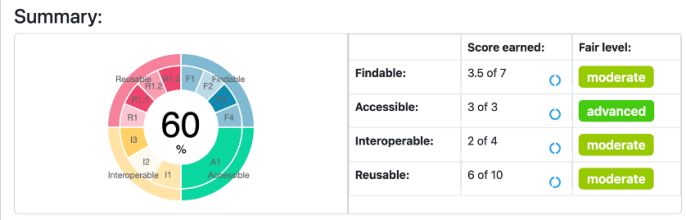


Fig. 3. FAIR evaluation report on the enhanced version

The implementation of these enhancements varies in difficulty but is crucial to improving the portal's alignment with FAIR principles. We briefly summarize the changes that will have the most impact on the FAIRness score:

1) Using established persistent identifier providers such as DOI, w3id, URN, or PURL is a key aspect of findability.
2) Ensuring that metadata explicitly includes core descriptive elements (creator, title, object identifier, publication date, publisher, object type, summary, and keywords) is another key aspect of findability.
3) Interoperability requires having multiple methods for metadata access. Including methods such as content negotiation, typed links, or SPARQL endpoints will improve the interoperability score.
4) Incorporating additional semantic resources such as DCAT2 which is a RDF vocabulary designed to facilitate interoperability between web-based data catalogs is a key area of improving interoperability.
5) Discrepancies between the file size in the metadata and the actual downloaded size impact the reusability score.
6) Detailed provenance tracking including the resource's editing history is another key factor in the reusability score.

While it can be argued that an overemphasis on the quantitative FAIRness score of a data portal takes away from the usability and domain or project-specific features and capabilities, we believe that having an agreed upon FAIRness evaluator and a community-driven implementation of capabilities that can ensure FAIR-compliant portals would be a valuable baseline for the science gateways community. While we do not claim to have solved this challenge, we hope that this paper provides some useful ideas for other gateway developers who need to design a customized data portal that solves a domain need, while also ensuring adherence to the FAIR principles.

Acknowledgment

This work was funded in part by NSF award no. 1835822. The portal was deployed to the Purdue Anvil Composable subsystem under the ACCESS allocation: EES220056.

References

[1] Mark D Wilkinson et al. "The FAIR Guiding Principles for scientific data management and stewardship". In: *Scientific data* 3.1 (2016), pp. 1–9.

[2] Robert Huber and Anusuriya Devaraju. "F-UJI: an automated tool for the assessment and improvement of the FAIRness of research data". In: *EGU General Assembly Conference Abstracts*. 2021, EGU21–15922.

[3] Rajesh Kalyanam et al. "GeoEDF: An Extensible Geospatial Data Framework for FAIR Science". In: *Practice and Experience in Advanced Research Computing*. PEARC '20. Portland, OR, USA: Association for Computing Machinery, 2020, pp. 207–214. ISBN: 9781450366892. DOI: 10.1145/3311790.3396631. URL: https://doi.org/10.1145/3311790.3396631.

[4] David G Tarboton et al. "HydroShare: advancing collaboration through hydrologic data and model sharing". In: (2014).

[5] Peter F Patel-Schneider. "Analyzing schema. org". In: *The Semantic Web–ISWC 2014: 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I 13*. Springer. 2014, pp. 261–276.

[6] Jean Brodeur et al. "Geographic information metadata—an outlook from the international standardization perspective". In: *ISPRS International Journal of Geo-Information* 8.6 (2019), p. 280.

[7] Ramanathan V Guha, Dan Brickley, and Steve Macbeth. "Schema. org: evolution of structured data on the web". In: *Communications of the ACM* 59.2 (2016), pp. 44–51.

[8] Kyle Chard et al. "The Modern Research Data Portal: a design pattern for networked, data-intensive science". In: *PeerJ Computer Science* 4 (2018), e144.

[9] Google. *Google Dataset Structured Data Documentation*. URL: https://developers.google.com/search/docs/appearance/structured-data/dataset.

[10] GeoEDF Project. *GeoEDF Metadata Extraction GitHub Repository*. 2024. URL: https://github.com/geoedf/geoedf-metadata.

[11] GeoEDF Project. *GeoEDF Portal GitHub Repository*. 2024. URL: https://github.com/geoedf/geoedf-portal.

[12] Anusuriya Devaraju and Robert Huber. *F-UJI - An Automated FAIR Data Assessment Tool*. URL: https://doi.org/10.5281/zenodo.6361400.

[13] FAIRSharing Authorship Group. *FAIRSharing FAIR Maturity Evaluation Tool*. URL: https://fairsharing.github.io/FAIR-Evaluator-FrontEnd/#!/.

[14] Daniel JB Clarke et al. "FAIRshake: toolkit to evaluate the FAIRness of research digital resources". In: *Cell systems* 9.5 (2019), pp. 417–421.

projectEureka
A Gateway for Cloud and K8s HPC & AI Bursting

Mary Brandenburg
Project Lead
Omnibond Systems
Central, SC
mary@omnibond.com

David Reynolds
Storage Developer
Omnibond Systems
Greenville, SC
David@omnibond.com

Aaron Crawford
UI Developer
Omnibond Systems
Anderson, SC
Aaron@omnibond.com

Jeff Denton
Infrastructure Developer
Omnibond Systems
Charleston, SC
JDenton@omnibond.com

Jeremy Grieshop
Infrastructure Developer
Omnibond Systems
Clemson, SC
jeremy@omnibond.com

Kristen Smith
Application Developer
Omnibond Systems
Anderson, SC
kristen@omnibond.com

Justin Cooley
Infrastructure Developer
Omnibond Systems
Pelzer, SC
Justin@omnibond.com

Boyd Wilson
CEO/CTO
Omnibond Systems
Salem, SC
boyd@omnibond.com

***Abstract*—A demonstration of projectEureka and overview of its design and development. It is based on a new meta-scheduler, omni-scheduler, and is built for routing HPC and artificial intelligence jobs between Kubernetes (K8s), multicloud, and traditional HPC schedulers. projectEureka leverages this unified meta-scheduler for interactive application integration with Open OnDemand with a newly developed project-based user interface.**

Keywords—HPC, Artificial Intelligence, AI, scheduling, Open OnDemand, Interactive Applications, Multicloud

I. Introduction

Drawing from our extensive experience in developing and maintaining CloudyCluster, we have encountered numerous scenarios where researchers and research computing support teams have articulated a need for enhanced integration between cloud-based and on-premises systems. This demand is accentuated by the ongoing expansion of computational workloads, which range from traditional batch High Performance Computing (HPC) and High Throughput Computing (HTC) to interactive computing, AI model training, and inference. These workloads span various platforms including Slurm clusters, Kubernetes clusters, and diverse cloud services, highlighting the critical need for tools that can streamline orchestration and usability of these complex environments.

To address this challenge, we initiated a project in January 2023 aimed at developing a system that tackles the following key problem areas in the current state-of-the-art: (1) Seamless cloud integration with on-premises systems, (2) Intelligent job routing between on-premises and multiple cloud accounts, (3) Integrated multi-point data staging between various cloud accounts and on-premises systems, and (4) A project-based user interface designed to simplify processes for researchers and support personnel.

In this paper, we discuss the design and development of a meta-scheduler-based system that incorporates these goals to provide a simple, secure layer for managing resources in a unified manner across both cloud and on-premises environments, as outlined in Fig 1.

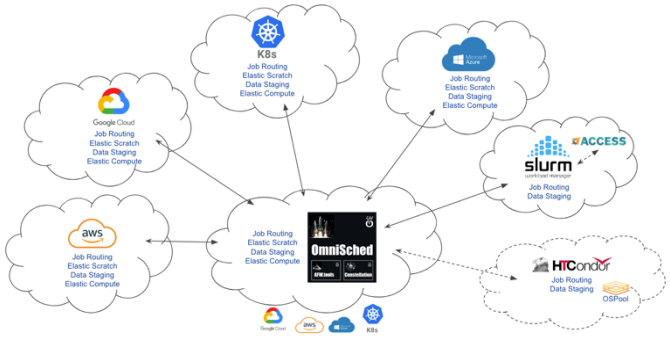


Fig 1.

II. Components

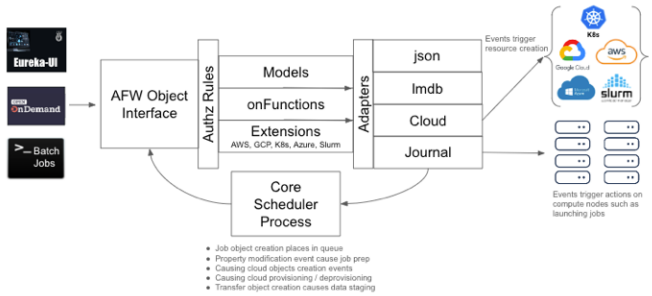
A. Meta-scheduler Core

The meta-scheduler will provide the core facility for managing jobs, job routing, multi-location resource provisioning, and simplifying the integration of core functionality, addressing the four key problem areas outlined in the introduction.

The core of the meta-scheduler architecture is built on Adaptive Framework (AFW)[1], which provides: (1) a JSON object store that surfaces a consistent interface for abstracted data-store access, (2) customizable rules-based authorization of object interactions, (3) internal and external event triggers based on object changes. Facilities have and will also be built around these to support: (4) an event driven data staging subsystem, (5) a comprehensive job language, and (6) an adaptable interface to support standard Slurm commands and additional job directives to provide access to the core capabilities.

As outlined in Fig 2, AFW provides a common interface to all the end user components, such as Open OnDemand[2], the Eureka project based user interface, and command line utilities, through https and leveraging JSON as the payload.

Fig 2.



These components, while authenticated as the end-user, reducing the risk of privilege escalation, can create objects that represent jobs directly into the meta-scheduler data-store. Authorization of this process is governed by the AFW rules that are put in place to securely restrict access to those previously given permissions within the object-store. Once the job object is created, an AFW event notification is sent to the meta-scheduler process which, based on the resource configuration, determines when and where the job should be executed. Once determined, the meta-scheduler will create the appropriate resource objects in the AFW-based data-store, triggering the resource creation in the respective provider. The providers will be either AWS, Google Cloud, Azure, Kubernetes (K8s), or in the case of job routing, it will forward the job to the designated slurm scheduler. When the meta-scheduler determines the resources from the provider are ready, an event-listener on the compute nodes will be notified of the job to be executed on the nodes. Whenever a job is marked completed it will cause an - to trigger the clean-up of the respective provider resources.

B. *Multi-cloud Data Staging System*

Seamlessly integrating data for research computing jobs is the third key area. Research data is expanding not only in size, but also in the number of storage locations. While there are tools such as Globus[3] that can help manage data between sites, within a site most users are left to leverage system utilities. After several months of attempting to use existing on-premises open tools for this, we determined that we needed to develop tools directly dedicated to on-premises multi-cloud data transfers.

A multi-point data transfer system, omni-copy (ocp), based on smart_open [4], has been developed to be leveraged by the meta-scheduler to handle data transfer as part of the job and resource provisioning process. As outlined above, the event driven process which translates jobs into the respective objects is core to how the meta-scheduler operates. An additional object-event that is built into the system is that of a data-transfer object. When a job script includes a transfer directive, it is translated into a data-transfer object that causes the specified data on a storage provider to be transferred to another location on its respective storage provider. The transfer will not initiate until all of the storage provider dependencies have been met, will be performed as the user to significantly reduce the likelihood of privilege escalation, and once the transfer is complete, will mark the data-transfer object as complete so the clean-up process can start when appropriate.

C. *Comprehensive Job Control Language*

Research computing workloads have been shell script jobs for many decades, and there is a great amount of experience and existing knowledge and technical infrastructure to support these. To enable simplified adoption we are creating meta-scheduler job directives that can be added directly to existing jobs that will enable them to leverage projectEureka to allow for elastic resource provisioning and job routing. To achieve this goal we will be creating a language that will allow for existing shell scripts, that contain the new directives, to be transformed into the more comprehensive language that can support multiple cloud and processing parts.

This job control language, Omni Control Language (OCL), has been designed to be able to take new OCL job directives that can be added to traditional slurm scripts and translate them into native OCL, thus making the meta-scheduler have day-one value to researchers. On the other side, OCL is designed to provide more flexibility and capability with regard to extensibility and more complex jobs, including cross-provider capabilities within a single job. The directives will be in four parts: (1) computing resources that will be provisioned on demand as the job requires, (2) storage resources can be provisioned on demand as the job requires, (3) data staging can be configured to stage data and results at the appropriate time during the job, and (4) allow for multiple script executions that can interact with the provisioned resources and staged data.

D. *Resource Provisioning Subsystem*

The resource provisioning subsystem is the event-driven interface that creates and destroys resources used by the meta-scheduler, including clouds such as AWS, Google, Azure, and Kubernetes. In the initial development versions of projectEureka, we built an interface called Constellation that leveraged Terraform as the cloud interface. We found that Terraform lacked the parallelism needed for the dynamic nature of research computing jobs. Terraform also did not provide real-time interactive status of the resources that it provisioned. Terraform also was not implemented with the most efficient scalable cloud resource choices, such as bulk inserts on Google cloud.

Once a majority of the AFW event-based system was in place we looked to leverage the same capabilities of AFW to provide provisioning and deprovisioning of cloud resources. AFW has customizable extensions, and we are in the process of building these extensions that allow the meta-scheduler to interact natively with the cloud-provider and Kubernetes APIs. This will enable real-time interaction with the cloud provider control planes and direct adaptive interaction with the cloud resources. Job routing between provisioned and existing systems will be handled based on events as well. The

interaction between the meta-scheduler and the resource will happen through a similar interface leveraging this AFW capability.

E. *Project Centric User Interface (UI)*

To simplify data access and interactive application launching capabilities, a project-based user interface was developed to allow project leads and members to share common project data sources and common applications. This project interface provides a common way to access self-service resources. The associated data locations can also be referenced for data staging activities mentioned previously.

The project-centric UI is built using Vue and UI5 web components. The overarching project overview page from the user interface can be seen in the screenshot in Fig 3.

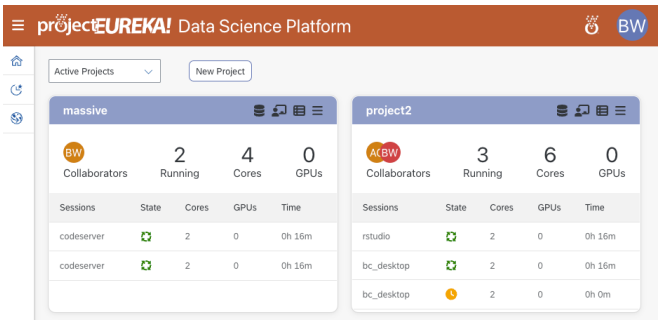


Fig 3.

An example of the running applications in a project is shown in the screenshot in Fig 4

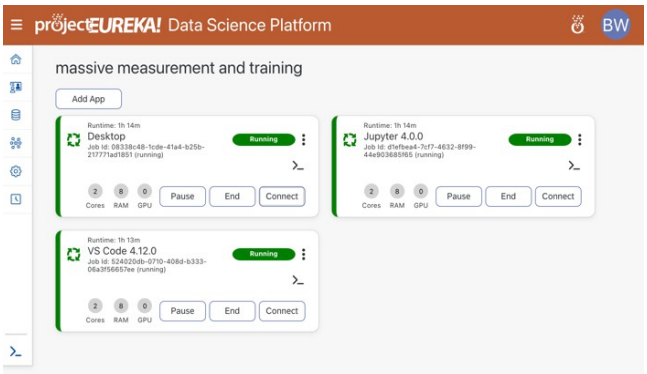


Fig 4.

F. *Multi-cloud Storage Management User Interface (UI)*

The storage manager UI provides a project-based view of the directories and files associated with a project so members can easily manually manage files as required between the various storage locations residing in different clouds and on-premise resources.

The data staging system is abstracted both from the UI and scheduler, allowing for common code and key management for the data transfer operations of data and results staging. The system allows for directory traversal and listing across multiple clouds and local resources in an extensible fashion, where it can easily be extended to support additional storage locations as outlined in Fig 1. An example of the storage manager UI is in Fig 5. This UI provides drag and drop support across multiple cloud and local storage resources. The storage manager UI leverages OCP outlined earlier.

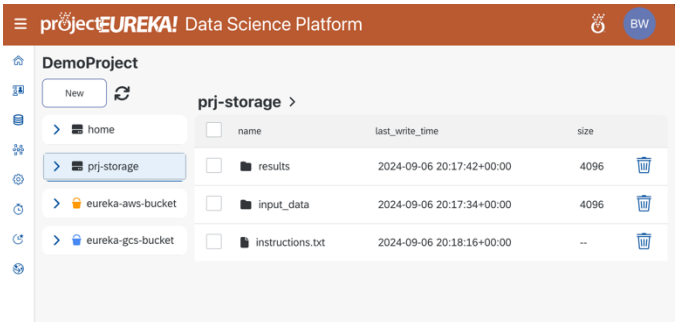


Fig 5.

III. Community Engagement

Throughout the months of developing projectEureka we have engaged with various communities to determine its usefulness and are tailoring it to meet the needs therein. We have had various meetings with potential communities in the US, EU, Japan, Singapore, and Australia; all have been met with positive feedback and significant interest in testing the system when available. The testing is scheduled to begin in the fall.

To address usability from a student community, we have participated in targeted ADMI and HPC in the City, SC23 hackathons. The system, projectEureka, was the main development collaboration system used in these hackathons, providing valuable usability feedback. As an example, the interactive job portion of the UI had previously launched separate cards from the launchers during application startup. After a majority of the 40 participants launched the same card multiple times we decided to have the main card transition to the running card and give another menu option to launch additional ones. We tested the changes during a design-safe hackathon with Texas Advanced Computing Center and it was much more user friendly.

To directly support the various research and education communities, the base of this project will be made free of charge with the option for paid support. We hope that this will provide a cross cloud and on-prem interconnectivity foundation for batch jobs, interactive virtual scientific workstations, and data staging that can be built upon.

As the final weeks of development approach on the v1 beta of this project, we have a list of users from various states and countries and different types of institutions, from technical schools to national centers, that will be deploying projectEureka to test, evaluate, and give feedback. Through this process we will iterate through the suggestions and incorporate ideas based on the feedback.

IV. Related Work

One of the concepts that is key to reducing costs of federated cloud infrastructure is to have a zero-cost footprint in the cloud as outlined in the Eric Lam Tapis paper [5].

Scalability of jobs in the cloud is required to meet the stringent demands of HPC and AI training systems as outlined in the Posey Urgent HPC papers [6] [7].

V. Conclusions and Work Breakdown

Given the broad interest in and need for integrating on-premises and cloud resources while providing integrated tools for data staging, elastic resource provisioning, and job routing and the growing complexity of computational resources, we feel this project has the potential to ease researcher computational interactions and reduce time to meaningful results.

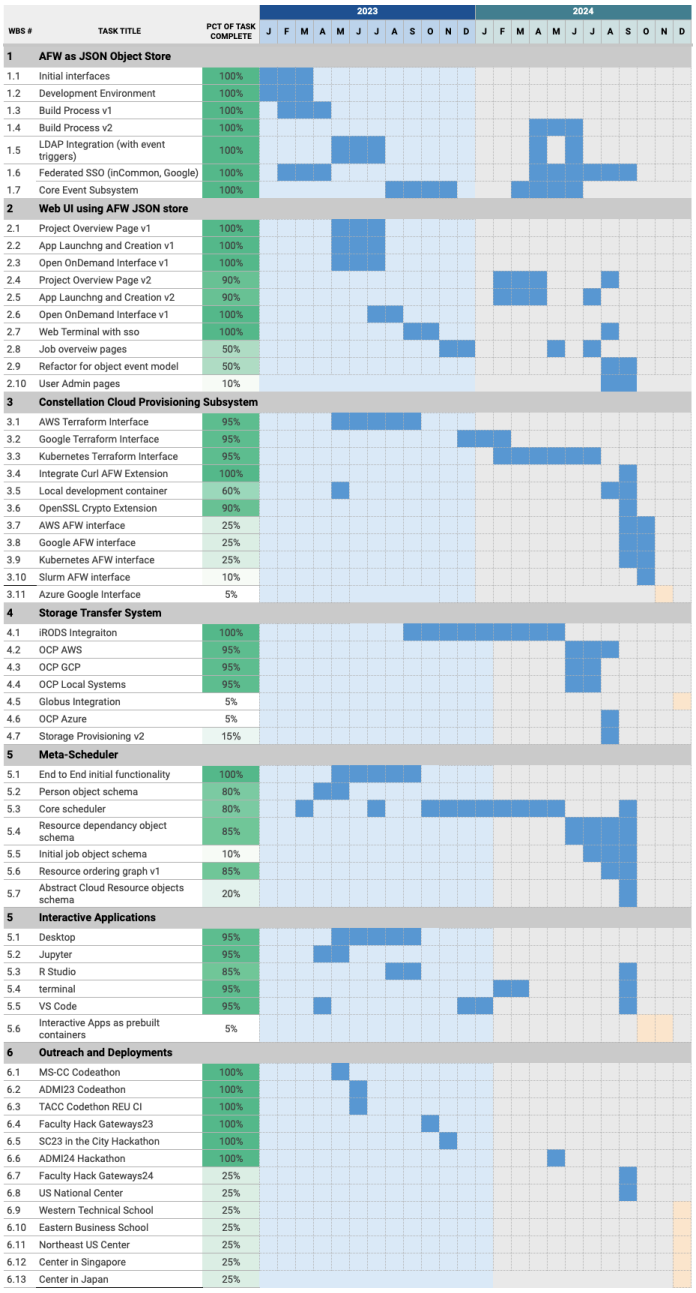
With our continued development and internal testing we are seeing that the simplified event-driven architecture is providing both good stability and scalability while reducing the active scheduler code, relying more on infrastructure pieces to accomplish the tasks at more efficient rates.

Additionally, based on the early usability tests with the platform at various hackathons it appears to accelerate access to resources and provides a time boost for those who use the system. Also, based on feedback from those hackathons, refinements have been made in the user experience.

Simplified data access through the web user interface also appears to streamline access and facilitate reduced time with data interaction.

We look forward to finishing the initial version of this project in the last quarter of 2024 and continue to iterate with the installations to refine, simplify, and improve the multicloud and on-premises research computing infrastructure integration.

The Gantt chart in Fig 6 outlines the progress to date and the anticipated workpieces through initial testing with sites that have expressed interest.



WBS #	TASK TITLE	PCT OF TASK COMPLETE
1	**AFW as JSON Object Store**	
1.1	Initial interfaces	100%
1.2	Development Environment	100%
1.3	Build Process v1	100%
1.4	Build Process v2	100%
1.5	LDAP Integration (with event triggers)	100%
1.6	Federated SSO (inCommon, Google)	100%
1.7	Core Event Subsystem	100%
2	**Web UI using AFW JSON store**	
2.1	Project Overview Page v1	100%
2.2	App Launchng and Creation v1	100%
2.3	Open OnDemand Interface v1	100%
2.4	Project Overview Page v2	90%
2.5	App Launchng and Creation v2	90%
2.6	Open OnDemand Interface v1	100%
2.7	Web Terminal with sso	100%
2.8	Job overveiw pages	50%
2.9	Refactor for object event model	50%
2.10	User Admin pages	10%
3	**Constellation Cloud Provisioning Subsystem**	
3.1	AWS Terraform Interface	95%
3.2	Google Terraform Interface	95%
3.3	Kubernetes Terraform Interface	95%
3.4	Integrate Curl AFW Extension	100%
3.5	Local development container	60%
3.6	OpenSSL Crypto Extension	90%
3.7	AWS AFW interface	25%
3.8	Google AFW interface	25%
3.9	Kubernetes AFW interface	25%
3.10	Slurm AFW interface	10%
3.11	Azure Google Interface	5%
4	**Storage Transfer System**	
4.1	iRODS Integraiton	100%
4.2	OCP AWS	95%
4.3	OCP GCP	95%
4.4	OCP Local Systems	95%
4.5	Globus Integration	5%
4.6	OCP Azure	5%
4.7	Storage Provisioning v2	15%
5	**Meta-Scheduler**	
5.1	End to End initial functionality	100%
5.2	Person object schema	80%
5.3	Core scheduler	80%
5.4	Resource dependancy object schema	85%
5.5	Initial job object schema	10%
5.6	Resource ordering graph v1	85%
5.7	Abstract Cloud Resource objects schema	20%
5	**Interactive Applications**	
5.1	Desktop	95%
5.2	Jupyter	95%
5.3	R Studio	85%
5.4	terminal	95%
5.5	VS Code	95%
5.6	Interactive Apps as prebuilt containers	5%
6	**Outreach and Deployments**	
6.1	MS-CC Codeathon	100%
6.2	ADMI23 Codeathon	100%
6.3	TACC Codethon REU CI	100%
6.4	Faculty Hack Gateways23	100%
6.5	SC23 in the City Hackathon	100%
6.6	ADMI24 Hackathon	100%
6.7	Faculty Hack Gateways24	25%
6.8	US National Center	25%
6.9	Western Technical School	25%
6.10	Eastern Business School	25%
6.11	Northeast US Center	25%
6.12	Center in Singapore	25%
6.13	Center in Japan	25%

Fig 6.

Acknowledgment

This work was supported by Omnibond, and the authors would like to thank the DICE lab at Clemson University for the prior research. The AFW development team that enabled us to create an efficient and flexible core. The OrangeFS project for the scalable storage system. The Open OnDemand team, the K8s community, Google Cloud, AWS, and Azure Cloud teams for their support.

References

[1] Grieshop, Gossett, Adaptive Framework Open Source Project, https://afw.tools.

[2] Hudak et al., (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.

[3] Foster, I., "Globus Online: Accelerating and Democratizing Science through Cloud-Based Services," Internet Computing, IEEE , vol. 15, no. 3, pp. 70,73, May-June 2011

[4] Radim Řehůřek et al., (2014), Smart Open - utils for streaming large files in Python, https://github.com/piskvorky/smart_open

[5] Lam, Eric, et al. "Extending Tapis Workflow Management Framework with Elastic Google Cloud Distributed System using CloudyCluster by Omnibond." Science Gateways 2022 (2022).

[6] B. Posey et al., "On-Demand Urgent High Performance Computing Utilizing the Google Cloud Platform," 2019 IEEE/ACM HPC for Urgent Decision Making (UrgentHPC), Denver, CO, USA, 2019, pp. 13-23, doi: 10.1109/UrgentHPC49580.2019.00008.

[7] B. Posey, et al., Dynamic HPC clusters within amazon web services (aws). Diss. Clemson University, 2016

Implementing Reproducible Cookbook Environments for Advanced Analyses on Science Gateways

Mobley
Texas Advanced Computing Center
University of Texas
Austin, USA
wmobley@tacc.utexas.edu

Pearson
Texas Advanced Computing Center
University of Texas
Austin, USA
0009-0006-4328-2476

Osorio
MetaLearn SPA
Chile
0000-0002-3611-6510

Tijerina
Texas Advanced Computing Center
University of Texas
Austin, USA

Trueheart
Texas Advanced Computing Center
University of Texas
Austin, USA

Faust
Department of Civil, Architectural, Environmental Engineering
University of Texas
Austin, USA
0000-0001-7986-4757

Pierce
Texas Advanced Computing Center
University of Texas
Austin, USA
0000-0002-3050-1987

***Abstract*—Science gateways face challenges supporting new users that require complex environments for High-Performance Computing (HPC). To address barriers to entry caused by complex environments, an application was developed to integrate science gateways with reproducible environments, reducing barriers to entry. Leveraging concepts from systems like Binder and Google Colab, we created reproducible computational cookbooks — a flexible framework that supports containerizing and sharing environments and workflows across users for advanced computing resources. These cookbooks utilize standardized repositories and Docker images, which simplifies the environment setup providing access through science gateways. By employing a developers' gateway, users can register and share applications to enhance collaboration. Once registered, applications facilitate collaboration in HPC research, ensures consistency across environmental dependencies, and enables access to reusable analytical workflows. This paper outlines the cookbook system's development, demonstrating its potential to broaden access and impact in HPC research environments.**

***Index Terms*—High-Performance Computing, Reproducibility, Science Gateways, Complex Environments, Workflow**

I. Introduction

Complex research questions often require difficult to install environments that can create barriers to entry for new HPC users. The Texas Advanced Computing Center (TACC),a leading academic HPC institution, currently has limited options to support users with low maintenance and reproducible environments. For many technically inclined users, this limited support is enough. Many research projects can build upon a foundation of previous HPC research projects. However, the amount of data available is ever increasing, and more users are needing access to higher performance computers. These users may not be comfortable with the command line interface or other technical aspects of currents. Science Gateways have been an answer to these technical problems. Science gateways provide a web based user interfaces without command line interactions, thus reducing barriers to entry [1], [2]. To date, TACC has lacked an effective workflow to connect the science gateway front end with standardized and reproducible environments. This paper discusses the workflow and application developed to provide this capability.

Various systems have been developed to improve scientific reproducibility by reducing required installation times and the variability within complex environments on scientific computing [3]. Systems such as Binder [3], Google Colab [4], and Open OnDemand [5] which provide a platform for creating the provided environments and running code, enabling users without extensive programming skills to rapidly interact with and utilize the tools without standing up and manage the underlying environment themselves. Other projects have focused on organizing training and environments specific to their field, for example Project Pythia for the geosciences [6], or using Google Colab for training economic students [7]. Data analysis workflows often use similar packages and the devel-

We acknowledge funding from the Planet Texas 2050 program of The University of Texas at Austin, the U.S. Department of Energy Office of Science, Biological and Environmental Research Program under Award Number DE-SC0023216, and the U.S. National Science Foundation Navigating the New Arctic Award No. #2127353.

opment of reproducible computational cookbooks provides the means to separate out the complex environment setup from the downstream data research work and enables the sharing of computational environments between collaborators and across projects. This improves reproducibility and transparency of research project. While effective, using cloud systems such as Google Colab or Binder at scale can be quite costly, especially if significant computational resources are needed. Many HPC systems leverage the Open OnDemand infrastructure, which allows the user to develop and share applications. However, TACC currently uses the TAPIS infrastructure an alternative system that provides a user-friendly and modern Web UI that best allows TACC's users to utilize our HPC systems without needing to use the command line. TACC's support for Academic research can significantly reduce the cost and make the work financially viable for users with access.

TACC users have access to powerful systems, such as Lonestar 6, but a number of constraints have limited user access to these resources. First, installing software includes navigating TACC security requirements which creates an extra layer of difficulty. Second, the compute nodes are hidden behind a login node which requires a customized JupyterLab [8] setup to run as an interactive web session [8]. A typical Jupyter install is feasible [9], however varying scientific domains require different components of the Jupyter environment, and as the environments become increasingly complex, reproducibility across users becomes more difficult. This is where the Binder project comes into play, the binder design provides a standard format to containerize repositories.

II. Current Status of TACC science gateways

The cookbook system is designed to support complex environments (Fig. 1). Historically, TACC has received fewer requests for certain packages, such as new software or geospatial libraries (ex. GDAL, OSGEO), but as this field has expanded further into the High Performance Computing space, users require more support for these packages and TACC has encountered issues providing these resources. Typically, when a package has a low request rate on the system, users are directed to install the package to their $WORK [10] space on the machines. However, when investigating how users might install these packages for themselves, we discovered that a complex process was involved in almost every installation. For environments such as GDAL, pip or conda install commands were insufficient because they rely on the presence of libraries the systems lack by default. Direct installation proves difficult as these packages' default to installing to the root space, a restricted space for TACC users. Most users seeking these packages were not Linux software specialists who were familiar with how to work around such barriers.

With these complex software, we initially settled on system installation by the administrators, but this approach required significant maintenance due to the number of dependencies a package (e.g. GDAL or NLP). As such, this solution requires frequent updates to all the associated software and libraries, and still, users need to modify their environments to ensure these packages are properly loaded in their $PATH before submitting their jobs. All of these issues require a level of HPC computing skill that creates a significant barrier to entry for newer users.

III. Use Case: Cookbook Solution

The initial use case for this project was the development of a cookbook environment that supports Natural Language Processing (NLP). NLP research provides a perfect test case for standing up this service on advanced computing resources. NLP requires complex computing installation and frequently involves researchers from non-technical backgrounds. These non-technical users lack the requisite programming skills to independently set up and maintain the necessary compute environments. Additionally, many projects of interest involve the analysis of potentially sensitive materials (example: unredacted interview data, critical infrastructure details) where maintaining control of the raw data, interim analytical data outputs, and resultant data products is critical.

The initial use case was implemented codesigning a solution with researchers on a Navigating the New Arctic research project (NNA Team). The NNA team uses mixed method approaches for evaluating the gaps between the socio-cultural understanding of water resources in remote, rural Alaska and the technical knowledge about the operational engineering specifications of the water system infrastructure that serves isolated communities. Using the cookbook service, TACC installed foundational large language models on HPC resources and setup a complex cookbook service to support multi-method NLP analyses that support the use of traditional NLP libraries, such as BERTopic, and emerging workflows that incorporate LLMs in a private environment. The social scientists and environmental systems engineers tested the cookbook systems using corpora from 1) interview data collections with rural Alaskan community members and 2) technical water infrastructure systems training materials. The NLP cookbook provides access to a new group of HPC users and helps complete advanced analysis was compared with manually coded results social scientists, and lead to the creation of a module in a software application called Sites and Stories. This module is in development and will provide AI-enabled support for modeling with stakeholders [11].

The cookbook services leverage science gateways and TAPIS to provide users complex environments for interactive scientific experiments. TAPIS is an API that connects to TACC allowing users to organize and submit jobs to HPC systems [12]. We adapted features from the Binder container standards to make the environments easy to develop from templates. While the cookbooks workflow can be used for a variety of environments, we focused on creating a flexible JupyterLab cookbook that can generate reproducible environments.

A. Required Repositories

The TACC cookbooks require two repositories to create a working application. The first repository [13] provides the Anaconda environment and the pip environment files in a

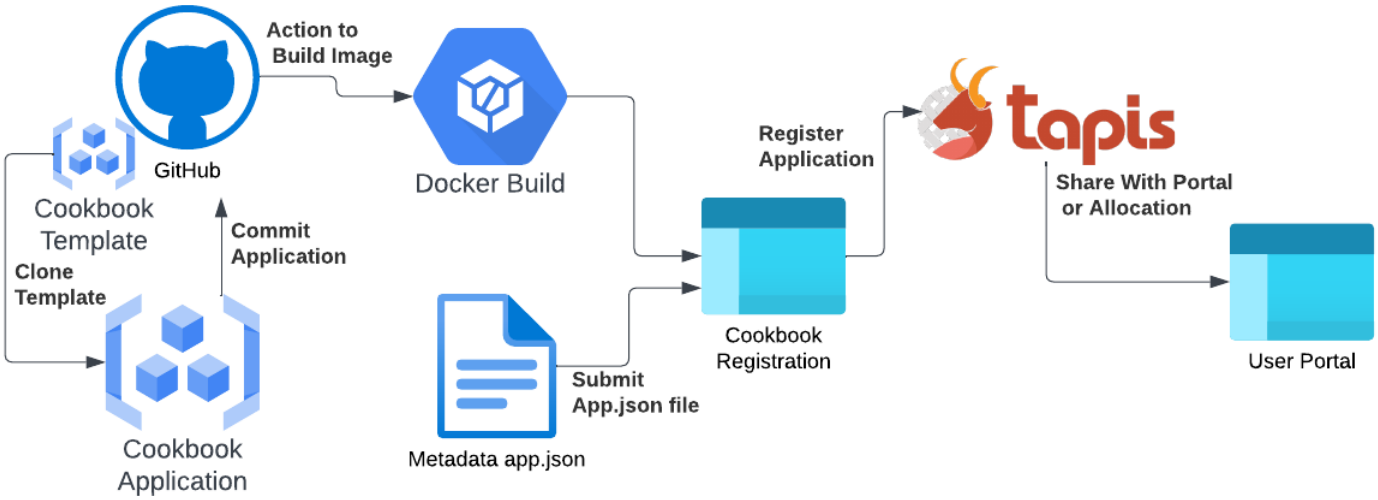


Fig. 1. Workflow and architecture for the TACC cookbook application. Users can 1) fork a template from github to create the appropriate repositories; 2) register and share the application with the science gateway.

format (stored in a ".binder" folder at the top of the repository) to work with Binder. Using this standard enables both files to be used for the anaconda environment. Note: the anaconda file is the primary environment file, while the pip file is only for dependencies that either (a) require large amounts of space or (b) are only available via pip. We separate the two environmental files to reduce the size of the cache required when creating the environment, as Pip allows for a no-cache install while Anaconda does not. This repository is pulled when creating the docker image for the container.

The second repository contains a Docker image forked from the TACC Jupyter Lab application notebooks [14]. The docker image sets up a JupyterLab environment, and then downloads the first repository to install the required dependencies. The application can further be configured to load and run additional software. By separating these two repositories, we can provide standardized repositories that require minimal customization for differing environments. This docker image has GitHub actions included in the repository to generate the docker image automatically.

Both repositories have been generated as templates living within GitHub.com. The current templates create a hello world JupyterLab the users can create on TACC systems. In the case of the NLP environment, we added a few functions that ensure the Ollama software [15] is installed and running as a service on the compute node. Ollama is a software that enables efficient loading and use of LLM models for Retrieval Augmented Generation (RAG [16]) of information within a corpora. Ollama provides an API that is queried through prompts, and is initiated through bash scripts. The NLP cookbook repositories provide examples of the working use case [17], [18].

B. Registering an Application

For development and production purposes, cookbooks need to be registered within the TACC systems. Previously, users would need to use TAPIS with either their own calls or using the Python API to register their application. For this project we developed a developers gateway [19] that allows users to register their application using a UI.

This gateway requires the user to register their app.json file with the gateway (Fig 2). The app.json file follows the TAPIS example and provides the list of choices a user can choose when submitting their application. This includes the queues that the application should use, and any other variables the image will need to run. For example, the NLP cookbook leverages GPUs for analysis, therefore the queues without GPUs have been removed from the drop down. Examples of these files can be found in the docker template repository [14].

Once the application has been registered the user can decide who to share the application with, whether this is a science gateway (ex. PTDataX [20]) or a specific allocation. By allowing the user to share to a specific allocation, we enable users to share environments in a collaborative manner, without publishing their application for all of TACC to access.

C. Availability on the Science Gateways

Once an application has been registered and shared it can become available on a science gateway. If the user is sharing with collaborators, the application will show up in the My Apps tab, however if the user wants to share it with the entire science gateway, they will need to request an admin add it to the cookbooks tab. Once the TAPIS cookbook application is running on a science gateway portal, the containerized cookbook allows the application to run across any execution system.

IV. Conclusion

Supporting users with complex software needs is a difficult problem, and a lack of this support at TACC has proved a significant research impediment for TACC users. In this paper we outline the process we have developed to reduce barriers to entry through cookbooks thereby increasing access and impact. The TACC reproducible computational cookbook services show promise for streamlining the reuse and sharing or workflows and providing users a variety of complex services on HPC systems without the need for root access. This workflow can be used both by science gateway managers, and also individual users to improve reproducibility and collaboration.

References

[1] Nancy Wilkins-Diehr, Michael Zentner, Marlon Pierce, Maytal Dahan, Katherine Lawrence, Linda Hayden, and Nayiri Mullinix. 2018. The Science Gateways Community Institute at Two Years. In Proceedings of the Practice and Experience on Advanced Research Computing (PEARC '18). Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/3219104.3219142 Received 26 April 2024, Vol. 1, No. 1, Article . Publication date: June 2024.

[2] Sean B. Cleveland, Rion Dooley, David Perry, Joe Stubbs, John M. Fonner, and Gwen A. Jacobs. 2018. Building Science Gateway Infrastructure in the Middle of the Pacific and Beyond: Experiences Using the Agave Deployer and Agave Platform to Build Science Gateways. In Proceedings of the Practice and Experience on Advanced Research Computing (PEARC '18). Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/3219104.3219151

[3] Project Jupyter, Matthias Bussonnier, Jessica Forde, Jeremy Freeman, Brian Granger, Tim Head, Chris Holdgraf, Kyle Kelley, Gladys Nalvarte, Andrew Osheroff, M Pacer, Yuvi Panda, Fernando Perez, Benjamin Ragan-Kelley, and Carol Willing. 2018. Binder 2.0 - Reproducible, Interactive, Sharable Environments for Science at Scale. In Python in Science Conference. Austin, Texas, 113–120. https://doi.org/10.25080/Majora-4af1f417-011

[4] 2024. Google Colab. https://research.google.com/colaboratory/faq.html.

[5] Chalker, Alan, Eric Franz, Morgan Rodgers, Trey Dockendorf, Doug Johnson, Doris Sajdak, Joseph P. White et al. "Open OnDemand: State of the platform, project, and the future." Concurrency and Computation: Practice and Experience 33, no. 19 (2021): e6114.

[6] Julia Kent, Drew Camron, John Clyne, Robert G. Ford, Maxwell Grover, Ryan May, Kevin Paul, Brian E. J. Rose, and Kevin Tyle. 2022. Project Pythia: A Pangeo Community Tool for Open-Source Education. 2022 (Dec. 2022), ED16B–06.

[7] Masanori Kuroki. 2021. Using Python and Google Colab to Teach Undergraduate Microeconomic Theory. International Review of Economics Education 38 (Nov. 2021), 100225. https://doi.org/10.1016/j.iree.2021.100225

[8] 2024. Jupyterlab/Jupyterlab: JupyterLab Computational Environment. https://github.com/jupyterlab/jupyterlab.

[9] Joe Stubbs, Julia Looney, Marjo Poindexter, Elias Chalhoub, Gregory J. Zynda, Erik S. Ferlanti, Matthew Vaughn, John M. Fonner, and Maytal Dahan. 2020. Integrating Jupyter into Research Computing Ecosystems: Challenges and Successes in Architecting JupyterHub for Collaborative Research Computing Ecosystems. In Practice and Experience in Advanced Research Computing (PEARC '20). Association for Computing Machinery, New York, NY, USA, 91–98. https://doi.org/10.1145/3311790.3396648

[10] 2024. Lonestar6 - TACC HPC Documentation. https://docs.tacc.utexas.edu/hpc/lonestar6/.

[11] Suzanne Pierce, William Mobley, Kasey Faust, and Keri Stephens. in preparation. AI-enabled Modeling with Stakeholders: Computational Infrastructure, Knowledge Capture and Reasoning. (unpublished).

[12] Joe Stubbs, Richard Cardone, Mike Packard, Anagha Jamthe, Smruti Padhy, Steve Terry, Julia Looney, Joseph Meiring, Steve Black, Maytal Dahan, Sean Cleveland, and Gwen Jacobs. 2021. Tapis: An API Platform for Reproducible, Distributed Computational Research. In Advances in Information and Communication, Kohei Arai (Ed.). Springer International Publishing, Cham, 878–900. https://doi.org/10.1007/978-3-030-73100_61

[13] Maximiliano Osorio, William Mobley, Lissa Pearson, and Suzanne Pierce. 2024. Cookbook Template Repository for Conda Environments. https://github.com/In-For-Disaster-Analytics/Cookbook-Docker-Template

[14] Maximiliano Osorio, William Mobley, Lissa Pearson, and Suzanne Pierce. 2024. Cookbook-Tutorial-Template. https://github.com/In-For-Disaster-Analytics/Cookbook-Tutorial-Template

[15] 2024. Download Ollama on macOS. https://ollama.com/download.

[16] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In Advances in Neural Information Processing Systems, Vol. 33. Curran Associates, Inc., 9459–9474.

[17] Maximiliano Osorio, William Mobley, Lissa Pearson, and Suzanne Pierce. 2024. Cookbook Template Repository for Conda Environments. https://github.com/In-For-Disaster-Analytics/Cookbook-Docker-Template

[18] Maximiliano Osorio, William Mobley, Lissa Pearson, and Suzanne Pierce. 2024. LLMRepository-Docker. https://github.com/In-For-Disaster-Analytics/LLMRepository-Docker

[19] Maximiliano Osorio and William Mobley. 2024. sites-and-stories-nlp. https://github.com/In-For-Disaster-Analytics/sites-and-stories-nlp

[20] Je'aime Powell, Sean Cleveland, Joe Stubbs, Suzanne Pierce, and Michael Daniels. 2019. Streamed Data via Cloud-Hosted Real-Time Data Services for the Geosciences as an Ingestion Interface into the Planet Texas Science Gateway and Integrated Modeling Platform.

Quakeworx science gateway: A custom instance of OneSciencePlace

Amit Chourasia, Choonhan Youn, Fabio Silva, Bar Olsen, Chunhui Zhao, Jeena Yun, Philip J. Maechling, David A. May, Ahmed E. Elbanna, Alice-Agnes Gabriel, & Yehuda Ben-Zion

Keywords: Science Gateway, Data management, Community models, HPC, Containers,

Earthquake rupture forecasts (ERFs) are critical seismic hazard research results that provide probabilities of future earthquake times, locations, and magnitudes for a given region. Several generations of ERFs have been developed for California jointly by the USGS and the Statewide California Earthquake Center (SCEC) and are used in broad impact seismic hazard maps. However, advanced physics-based models that account for fault system evolution and are well validated by observations are not yet used. Moreover, current models reside in the hands of a few highly skilled researchers, which reduces research pace and applications for societal benefits.

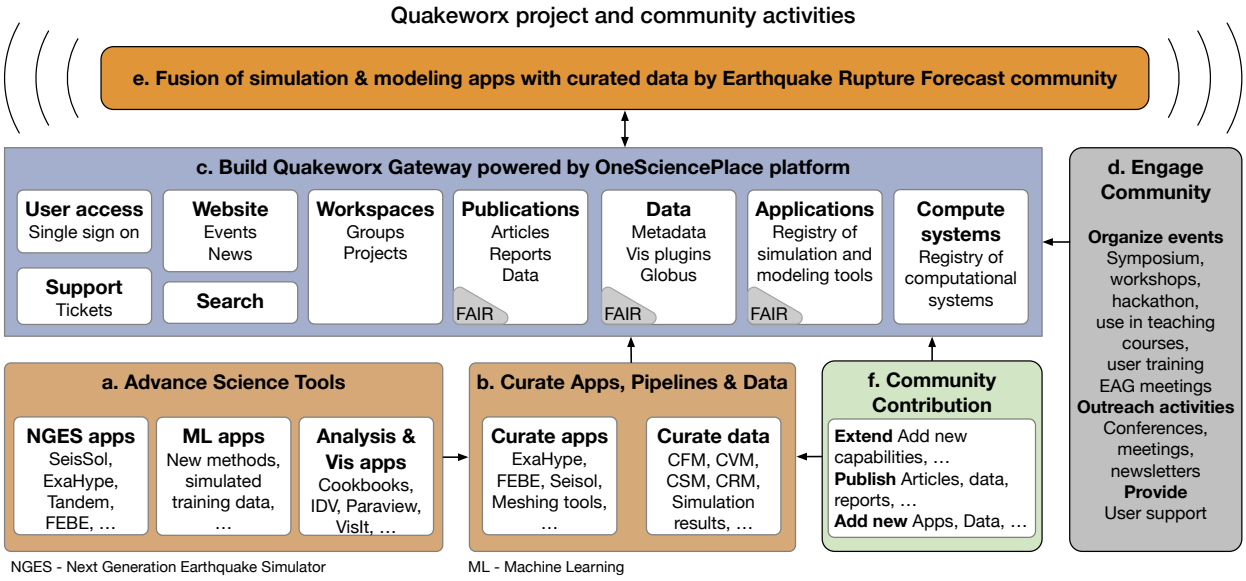


Figure 1: Illustration of Quakeworx activities: a) Project contributed apps to advance science tools; b) Curation of apps with input parameters and reference outputs to be deployed on gateway with user interface; c) Gateway that provides access to apps and data along with comprehensive set of collaboration, curation and execution capabilities; various research products can be published that follow FAIR data principles are highlighted with gray annotation; d) Engage community via multiple channels; e) Ability for the community to use the gateway for research, education and training; and f) Ability for the community to extend capabilities and publish research products including apps, data, and others.

The overarching goal of this project is to enable a wide community to access state-of-the-art models of rupture forecasts, further improve existing tools, validate model results, and use them for research and education. We have developed Quakeworx as a science gateway framework to significantly reduce barriers to execute/access simulation tools/data and facilitate rapid availability of emerging tools and results. Figure 1 illustrates different elements of the project. The Quakeworx gateway will accelerate innovation in earthquake science by enabling generation of diverse outputs (seismicity, ground motion, fault network configuration, strain rates, topography) that can be used to validate model results, improve ERFs, and discover new patterns.

Quakeworx gateway is built on OneSciencePlace, a content centric and composable online platform to transform delivery of FAIR content and computing in a single and easy to use environment.

The Quakeworx gateway framework will provide a range of capabilities that includes:

1. An app registry with a set of state-of-the-art-computational earthquake modeling codes such as SeisSol, Tandem, FeBE, RSQSim and others that are readily usable on computation resources. Both interactive (such as Jupyter) and command line apps are supported.
2. Curated data such as input configurations and output results for selected earthquake scenario simulations using various curated apps.
3. Online, web-based, workspaces for collaboration, training, and teaching. Apps, data and other content can be restricted to projects or shared with everyone.
4. Publishing capability that enables users to publish ancillary information such as reports, data and other content.
5. Users can easily use available apps, data, and publications or contribute new ones via a web browser. For instance, a new app can be contributed by uploading Docker or Singularity containers with necessary configurations and pairing them with the appropriate available computational resources.
6. A website that provides the above capabilities to all registered users via single sign-on from various institutions.

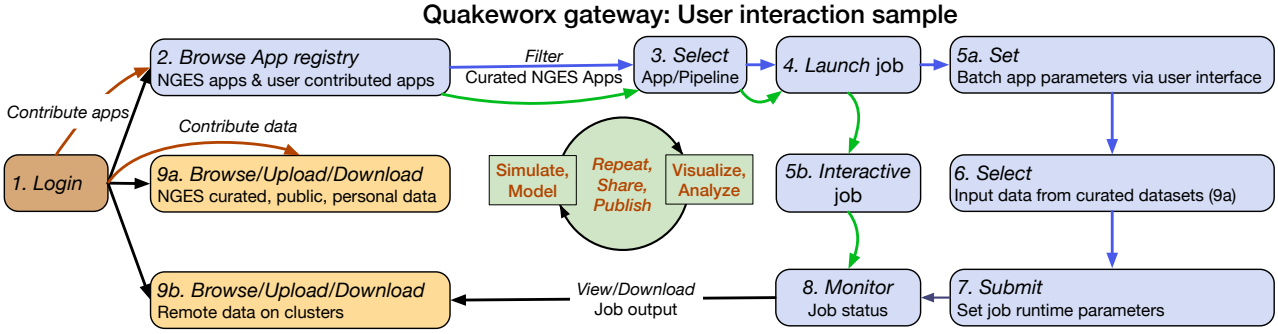


Figure 2: Illustration of sample user interaction paths on the gateway. Blue boxes depict a series of steps to perform simulation, modeling, visualization, or analysis. Steps 1–4, 5b, 8 & 9 with green arrows show an interactive session for an app. Steps 1-4, 5b, 6-9 with blue arrows show a batch job for an NGES app. Steps 1 & 9 show ability for users to contribute datasets for curation; apps may be contributed for curation with step 1 & 2 and both app and data can be published with FAIR principals. Yellow boxes indicate users ability to view and download curated NGES datasets, public or personal datasets (steps 1 & 9). The user interaction also allows cloning and repetition of simulation and analysis, and their output results may be shared or published at the gateway.

In this demonstration, we will provide an overview of the Quakeworx gateway. We will discuss apps, systems and data, single-sign-on capabilities and will also showcase some of the user interaction paths that are illustrated in figure 2. To ensure that it is accessible and easy to use for a large and diverse user community, the platform includes low-code and no-code options, as well as comprehensive training materials and tutorials. Our project's goal is to advance the science of rupture forecasts, to obtain new insights on earthquake physics, to improve seismic hazard estimates, and to create new opportunities for STEM education and engagement by providing a cyberinfrastructure platform for the broad community to easily use and reuse computational tools and data.

Quakeworx gateway: https://quakeworx.org
OneSciencePlace platform: https://onescienceplace.org

Enabling Workflow Performance Evaluation for Galaxy via Automated Benchmarking

Keith Suderman*, Nuwan Goonasekera†, Alexandru Mahmoud‡, Michael C. Schatz*, and Enis Afgan*§
*Department of Biology, Johns Hopkins University, Baltimore, MD, USA
†Australian BioCommons, University of Melbourne, Melbourne, Australia
‡Harvard Medical School, Boston, MA, USA
§Corresponding author: *enis.afgan@jhu.edu*

***Abstract*—As a science gateway grows, in terms of the number of users, number of jobs it handles, diversity of the workload, or number of installations, it becomes increasingly important to focus on efficiency. Efficiency helps make better use of available resources and hence better support users. Here we describe a benchmarking automation framework for the Galaxy science gateway platform that helps evaluate tool, workflow, and system performance. The framework, named ABM for Automated Benchmarking, allows users to easily benchmark and compare workflow runtimes on different Galaxy deployments as well as evaluate tool's performance using different resource configurations. Here we describe the ABM design, implementation details, and showcase its use in three difference usage scenarios.**

***Index Terms*—Galaxy, benchmark automation, cloud**

I. Introduction

Evaluating the performance of software tools and scientific workflows is crucial for end-users, tool developers, and system administrators. End-users need to estimate runtimes, costs, and compare performance across providers. Tool and workflow developers benchmark to optimize performance. Administrators evaluate for capacity planning and performance tuning. However, comprehensive performance evaluation is often time-consuming and nuanced, involving running tools with varying resource allocations, input parameters, and on different systems. For workflows, additional steps like tool installation and reconfiguration are required. Collecting and interpreting such performance data is complex, and therefore often overlooked, leading to sub-optimal user experience or system under-utilization.

We present Automated Benchmarking (ABM), a command-line tool that automates and tracks the benchmarking process of tools and workflows for Galaxy - a popular biomedical data analysis and workflow management platform [4]. Galaxy supports thousands of domain-specific tools and workflows, with public servers accommodating over 10,000 active monthly users and processing approximately two million jobs each month. Galaxy can also be launched on the cloud [7] or installed on local hardware. This scale and heterogeneity of uses necessitates performance optimization and cross-provider comparisons, motivating ABM's development. Other efforts, such as JUBE [6] and Hummingbird [2], are alternative benchmarking frameworks, but they lack support for Galaxy.

ABM enables comprehensive tool and workflow performance evaluation by automating the benchmarking process on any Galaxy installation, helping users better understand and optimize their usage. It handles uploading of workflows and input data, installing missing tools, running specified benchmark configurations, monitoring workload execution, and collecting runtime data. ABM uses descriptor files to simplify the definition of complex experiments. This approach also ensures consistent and repeatable benchmarking. We have used ABM to benchmark various analyses, demonstrating significant cloud cost reductions (40-80%) through identified configuration changes [1].

We envision ABM adoption during tool/workflow publishing, ensuring operational readiness, providing runtime estimates, and optimizing default resource configurations. In this paper, we describe ABM's capabilities, implementation, and usage, concluding with lessons learned and future enhancement ideas.

II. An overview of ABM

ABM enables users to run workflows, configure cloud compute resources, and collect job runtime metrics in an automated bulk fashion, thereby simplifying and accelerating the benchmarking process required for performance evaluation. ABM is implemented in Python with a focus on ease of use, scripting, and integration. It leverages BioBlend [8] and Planemo [3] to interact with the Galaxy platform (**Fig. 1**). ABM also includes mechanisms to monitor and manage the status of jobs, providing real-time feedback, job re-submission, and logs to users. ABM can be installed directly from PyPI and the source code is available on GitHub (*https://github.com/galaxyproject/gxabm*), where users can find documentation, access the latest updates, and contribute to development.

Interaction with ABM is managed via YAML-based configuration files that capture all the benchmarking steps while allowing users to define evaluation settings in a human-readable and editable format. This approach also allows details of each benchmark to be precisely captured, tracked, and easily shared. A sample set of definitions is shown in (**Fig. 2**) (for complete configuration details see the ABM documentation). The configuration files have a three-tier structure. The highest conceptual level is an *Experiment*. Each Experiment consists of one or more *Benchmarks* and each Benchmark consists of one or more *Workflows* and *Input Datasets*. The Experiment

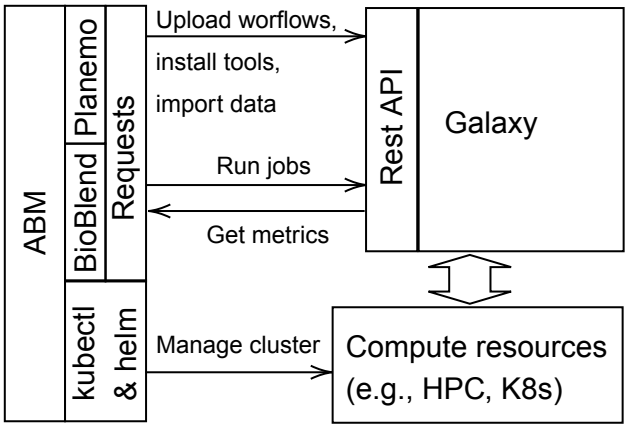


Fig. 1. Evaluation system design for ABM. ABM leverages existing libraries to interact with Galaxy and collect benchmark data. ABM can also reconfigure Galaxy to broaden the benchmarking space.

defines how many times each Benchmark should be run, which resources to use, and an optional set of resource configuration rules to iterate over. Each configuration rule defines the number of CPUs and amount of memory a tool should use. The Benchmark specifies input and output details for the Workflow, such as dataset and history names, and matches inputs from a Galaxy workflow submission form (**Fig. 3**). Workflow and Input Dataset definitions point to where those objects can be downloaded from.

A key advantage of using ABM to run workflows on several Galaxy servers is that users are able to use the human readable names for datasets, histories, and workflows. For security reasons, Galaxy represents these objects using ID values that are unique to a Galaxy instance. These ID values are used when interacting with the Galaxy server through the REST API. However, this presents problems when users want to run the same workload on several different Galaxy servers. ABM handles this translation from human readable name to the Galaxy ID, significantly reducing the setup and configuration time needed to perform benchmarking experiments.

III. Use Cases

In this section we present three sample use cases that each showcase ABM's utility for a different user persona. The first use case focuses on end users, allowing them to compare runtime characteristics of a workflow and performance of different Galaxy servers. The second use case focuses on tool or workflow developers and demonstrates how ABM can be used to benchmark workflow's execution as part of continuous integration. The third use case focuses on systems administrators, showcasing how to benchmark different cloud resource configurations.

A. Evaluate Workflow Performance on Different Servers

Currently there are more than 150 public Galaxy servers where various groups and institutions have set up a server

```
---
# Experiment definition
name: Variant-Analyses
runs: 3
benchmark_confs:
  - benchmarks/vc.yml
cloud:
  - australia
  - usa
---
# Benchmark definition
- output_history_base_name: Variant-Calling
  workflow_id: Generic variation analysis on WGS PE
    data
  runs:
  - history_name: 2GB
    inputs:
    - name: Paired Collection
      collection: Subsample of reads from
    SRR24043307
    - name: GenBank genome
      dataset_id: GRCh38.p14.gbff.gz
    - name: Name for genome database
      value: h38
---
# Workflows defintion
variant: https://benchmarking-inputs.s3.amazonaws.
    com/vc-iwc.ga
```

Fig. 2. A sample definition for an Experiment, a Benchmark, and a Workflow. The Experiment definition specifies some metadata, which Benchmark file to use, on which resources to run the Benchmark, and what job configurations should be used (job configurations require administrative privileges to use and are not shown here). The Benchmark definition specifies the Workflow and Input Dataset information. The Workflow definition specifies an online location of the Galaxy workflow file that ABM will import into the relevant Galaxy server. Input Datasets are defined in the same fashion as Workflows, but no example is shown. For a complete example, see *https://github.com/SchatzLabJHU/abm-examples*.

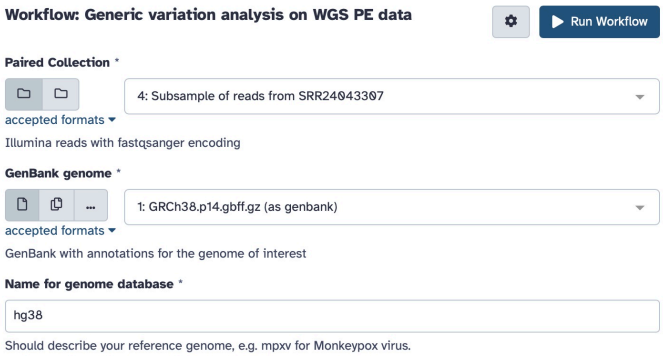


Fig. 3. The variant analysis workflow configuration form on the Galaxy server. The values and labels seen in the Galaxy interface are the same as the keys and values used in the Benchmark configuration file.

either as a public resource or a way to expose their tools: *https://galaxyproject.org/use/*. Within these servers there is a subset of servers known as *usegalaxy.**. These are general purpose, national Galaxy servers that cater to a wide user audience. Across all these servers there are many differences. These include systemic differences such as system capabilities, proximity to data, and legal jurisdiction. The differences also include configuration differences, such as which tools are

installed or how many resources those tools are configured to use. With many choices, service users often want to know which servers can run their workloads and which one perform best. Evaluating those servers is a time-consuming task requiring the user to transfer data and workflows as well as invoke and monitor job invocations. Here we showcase how ABM can be used to simply perform server comparison.

```
australia:
  url: https://usegalaxy.org.au
  key: ***User's Galaxy AU API key***
usa:
  url: https://usegalaxy.org
  key: ***User's Galaxy US API key***
```

Fig. 4. Example ABM profile configuration pointing to two Galaxy servers and letting ABM know how to access those servers. This is a global system configuration file that contains links to any number of servers.

Running the performance benchmarks requires us to set up the ABM configuration with the target servers. **Fig. 4** shows a sample configuration pointing to the Australian and US usegalaxy servers. With the necessary configuration available, we can invoke ABM to run a workflow on those servers:

```
#!/usr/bin/env bash
for server in australia usa ; do
    abm $server workflow import variant
    abm $server history import variant-2g
done
abm experiment run experiments/vc.yml
abm experiment summarize --markdown metrics/Variant-
    Anaysis
```

The script uploads a workflow and input data to the given Galaxy servers. Then the experiment is run (Experiment definition in **Fig. 2**). The source locations of the supplied workflow and input data objects from where they are uploaded to Galaxy are defined in their respective configuration files (not shown). Once submitted, ABM will monitor workflow execution and upon completion of all workflow jobs, ABM will retrieve runtime characteristics and summarize the results. A snippet of results is shown in **Table I**, capturing runtime characteristics of each tool in the workflow. The data shown in the table is a very condensed version of what the `history sumarize` command returns to make it fit the paper format. The full results returned include run ID, server, tool resource configuration, workflow name, Galaxy history name, inputs, tool ID, tool version, job state, number of CPUs allocated, amount of memory allocated, job runtime, number of CPUs used, memory limit, and maximum memory usage. The output from this command can be saved to a CSV file for downstream analysis or a more compact Markdown format for a human readable overview.

As can be seen, with ABM, once a Benchmark has been configured, the benchmarking process is reduced to a handful of commands, making it a straightforward task for users. Once the results are obtained, the evaluation can be performed. At the moment, this is done by other downstream tools, such as a spreadsheet, although there is future work planned (described below) to interpret the results as well. Also, the next section shows an example of how short scripts can be used to aggregate and interpret the collected data.

Run #	Tool	Runtime (Sec)	Memory (GB)
1	`lofreq_indelqual`	154	2.759
2	`lofreq_indelqual`	156	2.759
2	`samtools_stats`	40	0.005
	...more results...		

TABLE I
EXAMPLE OUTPUT GENERATED BY THE ABM *history summarize* COMMAND, ALTHOUGH HEAVILY CONDENSED TO FIT THE PAPER FORMAT. THE OUTPUT OF THIS COMMAND ALLOWS A USER TO EASILY GET A IDEA OF RUNTIME CHARACTERISTICS OF INDIVIDUAL TOOLS COMPRISING A WORKFLOW. THE INTENT IS TO USE THIS DATA IN DOWNSTREAM TOOLS FOR FURTHER ANALYSIS AND VISUALIZATION.

B. Monitor Workflow Performance with Continuous Integration

Galaxy has sophisticated workflow capabilities with complex control structures, such as conditionals, dataset collections, and sub-workflows. These features are exposed via an accessible graphical editor, making it straightforward to develop sophisticated workflows. In addition, there are nearly 10,000 tools available in the Galaxy ecosystem, often offering several choices for accomplishing a given step of the data analysis process. As a consequence, design decisions and tool choice may have impact on workflow performance and runtime characteristics. As workflow developers interact with this increasingly complex system, there is a growing need to regularly evaluate and compare performance of different workflow builds. Ideally, this evaluation is part of the development and publishing life cycle.

Here we provide template GitHub repository with a GitHub Action that leverages ABM and allows a workflow developer to easily run workflow benchmarks and collect performance data (*https://github.com/SchatzLabJHU/abm-action*). The GitHub Action (shown in **Fig. 5**) will upload a given workflow and test data on the specified Galaxy servers, monitor its execution, collect the resulting performance data, and deposit the data in the repository. The Action can be defined to automatically run on each commit or pull request, making it a seamless and integral part of workflow development process. The collected performance data is visualized in the repository as default preliminary analysis while additional evaluation is certainly possible in downstream tools.

To start using this capability, anyone can add the action to a GitHub workflow, do a one-time configuration to supply the Galaxy API keys, and update a link to the workflow file and its input data. Subsequently, every update to the workflow file will yield a performance benchmark data point allowing easy evaluation of any performance changes.

C. Estimate Cloud Costs

The third use case showcases ABM's full capabilities in identifying the most desirable resource configurations for a given workload, going beyond benchmarking existing servers or ensuring workflow operability. ABM can reconfigure

```
- name: Test a workflow with ABM
  uses: SchatzLabJHU/abm-action@v1
  with:
    org-api-key: ${{ secrets.ORG_API_KEY }}
    histories-path: histories.yml
    workflows-path: workflows.yml
    benchmarks-file-path: benchmarks/vc.yml
    experiments-file-path: experiments/vc.yml
```

Fig. 5. GitHub Action used for Continuous Integration testing of a workflow.

Galaxy to explore how configuration changes impact workload performance. We leveraged this capability to benchmark cloud resources across tool/resource configurations to estimate workload costs on commercial cloud providers [1]. Cost estimation is invaluable for scientists considering cloud adoption, where cost uncertainty remains a major hurdle.

To identify desirable resource configurations and estimate the costs, we performed a number of experiments to measure the performance of several workflows across three dimensions: input data size, number of CPU cores allocated, and amount of memory allocated for a tool. In addition to all the benchmarking steps already described above, this use case requires Galaxy to be reconfigured for each distinct benchmark. Changes include updating Galaxy's configurations to specify desired amount of resources for given tools and restarting necessary processes. Because these are system-level changes, this mode of ABM works only on Galaxy servers deployed on Kubernetes and using the Galaxy Helm chart [5]. Kubernetes offers a reliable mechanism for performing such changes in an automated fashion and is hence the reason why ABM only supports cluster changes for Kubernetes deployments.

To run such evaluation, in addition to defining an ABM Experiment, such as the one in in **Fig. 2**, it is also necessary to define desired job configurations as part of the Experiment:

```
job_configs:
  - 8x32
  - 32x64
```

Each of these configurations maps to a file that contains necessary rules indicating how many resources should be allocated for a given tool (e.g., *https://github.com/ksuderman/Benchmarks/blob/master/rnaseq/rules/8x32.yml*). Behind the scenes, ABM issues *kubectl* and *helm* commands to make the necessary cluster changes, updating the Galaxy deployment, and then instantiating the suitable workload. One thing to keep in mind that ABM will not scale the underlying cluster, which needs to have adequate resources to accommodate desired job configurations defined in the Experiment. In this mode, ABM can also install any missing tools in Galaxy, further automating the benchmarking process for newly created or temporary Galaxy deployments.

IV. Discussion and Future Work

A systematic approach to collecting benchmarking data is crucial for meaningful performance evaluation. The ABM tool streamlines this process with its well-defined, reproducible, and shareable benchmarking configurations. This allows users from various backgrounds to easily evaluate their workloads. Our use cases validate ABM's capabilities and demonstrate its support for different user personas.

One of the most impactful lessons from working with ABM is the importance of thoughtful benchmarking design. While ABM's design capabilities enable quick execution of numerous jobs and collection of extensive runtime data, this can overwhelm users and produce noisy results. It is essential to focus on the specific goals of the evaluation and design compact experiments that test clear hypotheses, rather than amassing large volumes of data without direction.

ABM enhances users' ability to understand correlations between different configuration options, such as resource requirements relative to input data size. Despite its current automation capabilities, manually collecting, organizing, and interpreting the resulting data can still be challenging. To address this, we plan to develop a hosted ABM service. This service will allow users to upload an Experiment configuration and ABM will automatically run the necessary workloads, generating a summary report that highlights key findings.

In conclusion, ABM provides a powerful tool for systematic and reproducible performance evaluation of Galaxy workloads, enabling users to optimize their workflows. Our future enhancements aim to further simplify the benchmarking process, making it more accessible and informative for users.

Acknowledgment

We would like to thank the Galaxy Community for their contributions to the platform. This work was supported, in part, by NIH awards U24HG010263, U24HG006620, and R03CA272952 as well as NSF award 2005506.

References

[1] Enis Afgan, Keith Suderman, Nuwan Goonasekera, Bridget Carr, Victor Wen, Peiyuan Xu, Michelle Savage, Tyler Collins, and Michael C. Schatz. Estimating cloud computing costs for bioinformatics workloads. *[Manuscript in preparation]*, 2024.

[2] Amir Bahmani, Ziye Xing, Vandhana Krishnan, Utsab Ray, Frank Mueller, Amir Alavi, Philip S Tsao, Michael P Snyder, and Cuiping Pan. Hummingbird: efficient performance prediction for executing genomic applications in the cloud. *Bioinformatics*, 37(17):2537–2543, 2021.

[3] Simon Bray, John Chilton, Matthias Bernt, Nicola Soranzo, Marius van den Beek, Bérénice Batut, Helena Rasche, Martin Čech, Peter JA Cock, Björn Grüning, et al. The planemo toolkit for developing, deploying, and executing scientific data analyses in galaxy and beyond. *Genome research*, 33(2):261–268, 2023.

[4] The Galaxy Community. The Galaxy platform for accessible, reproducible, and collaborative data analyses: 2024 update. *Nucleic Acids Research*, 05 2024.

[5] Nuwan Goonasekera, Alexandru Mahmoud, Keith Suderman, and Enis Afgan. Galaxy Helm chart: a standardized method for deploying production Galaxy servers. *Bioinformatics*, 40(8):btae486, 08 2024.

[6] Sebastian Lührs, Daniel Rohe, Alexander Schnurpfeil, Kay Thust, and Wolfgang Frings. Flexible and generic workflow management. In *Parallel computing: On the road to exascale*, pages 431–438. IOS Press, 2016.

[7] Michael C Schatz, Anthony A Philippakis, Enis Afgan, Eric Banks, Vincent J Carey, Robert J Carroll, Alessandro Culotti, Kyle Ellrott, Jeremy Goecks, Robert L Grossman, et al. Inverting the model of genomics data sharing with the nhgri genomic data science analysis, visualization, and informatics lab-space. *Cell Genomics*, 2(1), 2022.

[8] Clare Sloggett, Nuwan Goonasekera, and Enis Afgan. BioBlend: automating pipeline analyses within Galaxy and CloudMan. *Bioinformatics*, 29(13):1685–1686, 2013.

An Introductory Guide to Developing GenAI Services for Higher Education

Sarah Rodenbeck
Rosen Center for Advanced Computing
Purdue University
West Lafayette, IN, USA
srodenb@purdue.edu

Erik Gough
Rosen Center for Advanced Computing
Purdue University
West Lafayette, IN, USA
goughes@purdue.edu

Ashish
Rosen Center for Advanced Computing
Purdue University
West Lafayette, IN, USA
ashish@purdue.edu

Sathvika Kotha
Rosen Center for Advanced Computing
Purdue University
West Lafayette, IN, USA
kotha8@purdue.edu

K. Meher Hasanth
Rosen Center for Advanced Computing
Purdue University
West Lafayette, IN, USA
kmeherha@purdue.edu

Durga Dash
Rosen Center for Advanced Computing
Purdue University
West Lafayette, IN, USA
dashd@purdue.edu

***Abstract*—This paper reports on the lessons learned from developing and deploying campus-wide large language model (LLM) services at Purdue University for generative AI (GenAI) applications in education and research. We present a framework for identifying an LLM solution suite and identify key considerations related to developing custom solutions. While the GenAI ecosystem continues to evolve, the framework is intended to provide a tool- and organization-agnostic approach to guide leaders in conversations and strategy for future work and collaboration in this emerging field.**

***Index Terms*—Large Language Model Deployment, GenAI, Cloud Computing, Software Design, Requirements Analysis, Information Retrieval, Language Models, Software Management**

I. Introduction

The release of recent Large Language Models (LLMs) such as ChatGPT, LLaMA, Claude, and Gemini has catalyzed significant interest in Generative AI (GenAI) across various fields, including education [1]. The University of Michigan has been at the forefront in this area, notably creating and deploying the U-M GPT and U-M Maizey tools, designed for general use and fine-tuning, respectively [2].

Over the past six months, our team at Purdue University has sought to assemble a suite of LLM services to satisfy a range of use cases within the campus and ACCESS communities. Through our exploration, we have realized that there is little agreement over best practices and approaches to embarking on similar projects as new tools are released so frequently [3], [4]. Additionally, contrary to traditional software projects, a key element of GenAI projects is adapting to constant changes in the landscape (both internal GenAI-related priorities/appetite and external services available). Therefore, planning to pivot is a top-level consideration. This paper shares our experiences and learnings in evaluating, adapting, and deploying a suite of LLM services to enable AI-powered research. We have sought to leverage this into a general framework with guidance for other groups embarking on similar projects.

II. Methods

We present our approach in the context of a decision framework, which includes requirement analysis, suite selection, and custom service development considerations as top-level issues, which are discussed in turn.

A. Requirement Analysis

One pitfall that early adopters may fall into is to assume that finding a single approach that works for all the needs at a university is the single high-priority first step. On the contrary, a suite of services is the likely outcome of a substantially developed GenAI initiative. To note, the outcome might still converge to one or a few solutions depending on the mission or size of an institution; however, by assuming a multi-pronged approach from the start, the development effort will likely be more adaptable as requirements change. Therefore, much of the early effort is best spent on formalizing 1) overall project constraints, 2) user profiles, and 3) intra-project prioritization rather than searching for a single solution. While a general requirement analysis approach is still valid, there is a subtle difference in the benefits posed by conducting this type of analysis within the GenAI space. Rather than only being used for initial planning, such an analysis is used as a basis for assessing field developments. For example, suppose a new model or tool comes out. What is the relative benefit of adding support for this into the original suite versus shifting to support this instead of something in the original suite versus doing nothing and keeping the original approach? This assessment depends on fully understanding the vision and resources behind the project and the prioritized user profiles but allows more focused discussions.

B. Tool Selection

This part of the framework primarily concerns identifying a solution suite that satisfies the prioritized use cases from section 2A. It explores the balance between commercial and

custom services, addressing not just a simple build vs. buy decision but the optimal mix for usage and guidance.

Control is an overarching theme in this analysis, focusing on the control provided by various solutions as well as the level of control required. For instance, freely available tools such as the public version of ChatGPT pose a trade-off with data security, with chat histories being used by default for further fine-tuning and offering users very little control [5]. Our community's regular interactions with proprietary data and the need for privacy and IP considerations made such solutions generally inappropriate for organizational GenAI use in our case.

Commercial services like Azure AI Studio offer access to proprietary models while offering additional data privacy [6] but remain bound to the provider's policies and pricing structures. These services necessitate the use of commercial cloud resources and do not allow for the extraction of fine-tuned models. However, they support a broader range of models and include low-code tools for creating personalized "Assistants" with custom data. These services' features have been converging, with most providing APIs, model playgrounds, assistant-building tools, and user data support. However, different services offer access to different proprietary models (e.g., Gemini is only available through Google). Thus, partner selection hinges on existing organizational partnerships, desired flexibility/model access, and cost considerations. Cost estimations consider factors like deterministic seat-based versus undefined usage-based fees, intended usage, and the size and characteristics of the user base (e.g., API calls vs interface) [7], [8]. To serve all researcher needs (e.g., if access to many different proprietary models is needed), forming partnerships with multiple commercial services may be necessary.

Custom services, built from open-source tools/models and hosted with on-premises resources, require more resources to deploy and maintain but offer greater control. This makes them particularly appealing to power users or those with stringent data protection needs. Despite the growth in available open-source tools, most resources still lack complete functionality out of the box. Additionally, for the goals of our project, building a custom service was crucial to enabling lower-cost or free access to Generative AI tools to users who may not have the funding to pay for commercial services. Rather than having usage-based costs, custom services will incur significantly more development and infrastructure costs, which are discussed in more depth in section 2C. Still, when such resources are already available, this can essentially subsidize the raw cost [9].

Mapping use cases to the necessary level of control, balanced against costs and priorities, forms a strategic execution roadmap. It is crucial to understand both the short-term and long-term solution landscapes. Figure 1 illustrates a sample mapping of user types to services and feature usage, highlighting the flexibility in these mappings. While enterprise cloud services may require less startup time, the end state of custom services can be comparable, mainly differing in the models available (which may be a key consideration, depending on the user base). Initially, a GenAI solution suite might focus more on enterprise services, but users may be guided toward custom services as new features are added. Additionally, some scenarios may necessitate hybrid solutions where custom features are developed on top of commercial APIs.

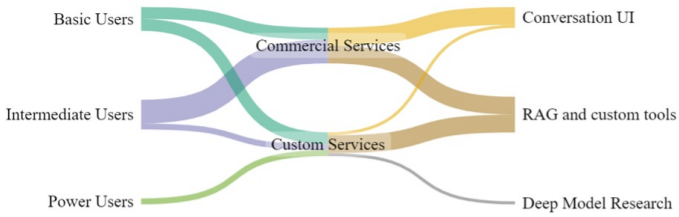


Fig. 1. Mapping of users to services and feature usage.

C. Custom Service Development

Assuming that a custom service has been identified as part of the solution suite, as we did, understanding what goes into a custom service is also critical for planning. We conducted a comprehensive investigation into the process of deploying an on-prem LLM, specifically the LLaMA LLMs, onto the Anvil composable subsystem—a Kubernetes-based private cloud for deploying and managing persistent services [10].

1) Approach and Deployment: The massive size of LLMs, such as LLaMA3 with 70 billion parameters [11] and GPT 3.5, the most recent OpenAI model for which details are available, with 175 billion parameters [12], makes deployment more complex than with smaller services. Initial issues included errors related to insufficient CUDA memory–internal testing showed that hosting the full LLaMA3 model required 4x 80GB GPUs. Employing quantization was paramount to navigating this issue. Quantization involves some information loss, in this case reducing the precision of numerical values, but allows the model to fit into smaller hardware than the cluster it was trained on [13]. Significant work has been done on these topics, and tools like Ollama automatically employ quantization methods, even enabling LLMs to run on local machines [14]. However, running quantized models on a cluster with GPU resources improves performance. Initially, we used a more manual approach for quantization, but the development ecosystem surrounding Ollama presented a very attractive option.

However, the model must still be paired with sufficient GPU memory and resources to work well, especially for a large community of users. On average, model performance correlates with model size, but smaller models may still be sufficient for particular tasks [15]. Assuming that anticipated users will want a choice of models, though, sizing the deployment to account for the largest model size will result in the best performance. While LLaMA 3 70B, for example, does work on GPUs with smaller memory, response times

suffer. For satisfactory performance with multiple models, we recommend the minimum resource request includes 16 CPUs, 32GB of RAM, and a GPU with 40GB of memory. 40 GB of GPU memory allows for loading multiple small models simultaneously as well as larger models like LLaMA 3 70B with 4 bit quantization. Deploying the solution in an auto-scaling platform like Kubernetes is also desired to handle the highly variable GenAI resource demand. Ongoing computing costs must be factored into budgetary discussions.

2) Performance Benchmarking: Beyond our basic assessment of minimum resources, benchmarking performance was an essential aspect of our work to validate the scalability and limits of our approach. While the actual models and tools continue to shift, we anticipate the benchmarking approach to be more transferrable [16].

Our approach was based on LLMPerf [17], a library designed to validate and benchmark the throughput and latency performance of LLMs [18]. The library supports multiple backend frameworks and provides customizable scenarios to simulate real-world usage. Key features include scalable testing, detailed performance metrics, and extensibility for integration with existing testing infrastructures. LLMPerf's design also allows users to reproduce results reliably, ensuring that performance assessments are accurate and repeatable. We found it particularly valuable for optimizing LLMs for specific applications, as it offers pre-configured and user-defined benchmarking capabilities. Building on LLMPerf's framework and what we have found to be standard metrics in this domain, we chose to assess performance based on the following metrics.

- Inter-Token Latency: Displays variability in processing time between tokens, assessing fluctuating token processing speeds [19].
- End-to-End Latency: Total time from initiating to completing a request, pointing to variability in total response times.
- Time to First Token (TTFT): Latency from starting a request to generating the first token, where differences can indicate potential variations due to system load or complexity.
- Request Output Throughput: Measures tokens generated per second, which helps understand the system's efficiency in producing output.

We used a containerized benchmarking environment to ensure a stable and replicable platform [20]. Benchmarking was performed on an Ollama v0.2.4 container instance running in Kubernetes that was allocated 32 CPUs, 64GB of RAM and one A100 80GB GPU. The instance was configured to process up to 64 parallel requests using the OLLAMA_NUM_PARALLEL configuration option.

While LLMPerf did not have native support for an Ollama model backend, we were able to use LiteLLM, a library that standardizes API interactions across various LLM providers, to facilitate interaction [21]. Notably, LiteLLM's design is optimized for low-resource environments, making it particularly beneficial for our containerized setup, where resource efficiency is paramount.

We used a dedicated node on Purdue's Negishi cluster with 128 cores and 256 GB RAM as our benchmarking client and validated no client bottlenecks were present. LLMPerf load tests were conducted from inside a conda virtual environment for an increasing number of parallel requests.

While results will be highly dependent on the allocated resources for an LLM deployment as well as parameters like number of input and output tokens, a sample of our results can be found in Figure 2. Unsurprisingly, latency reliably increases with the number of concurrent requests. The relationship between Inter-Token latency and End-to-End latency is also easily visualized. Results for token throughput show that throughput drops quickly as more parallel requests are handled, eventually reaching a point where responses are less than 10 tokens/s, a threshold that would be considered too slow for users of the system. Using this benchmarking data, we plan to do future work to determine an optimal auto-scaling configuration for Ollama instances.

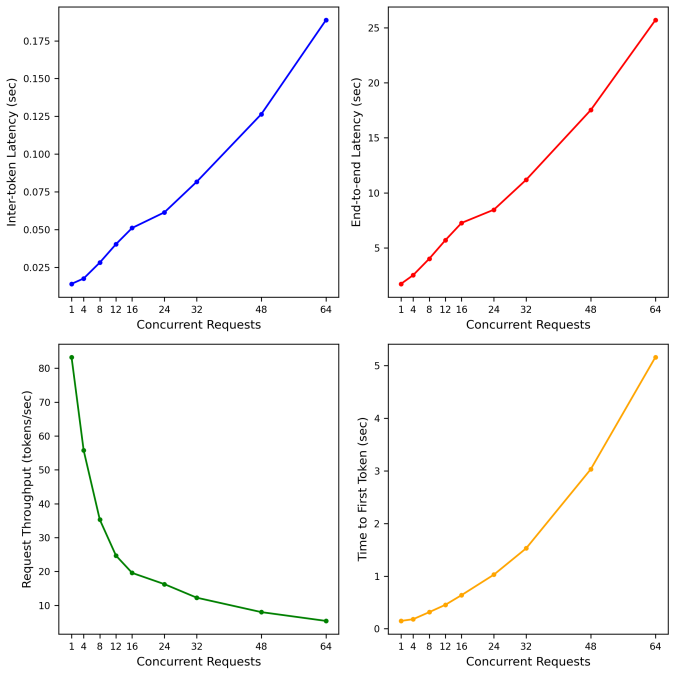


Fig. 2. Benchmarking Results

3) Additional Development Considerations: Beyond model deployment, numerous features are required to make custom services usable to a community. Pursuing a hybrid solution for building custom features on top of commercial model APIs may be possible but typically requires configuration as well [25]. However, solutions like Ollama are approaching the functionality offered by commercial solutions by automatically handling features such as streaming responses, not just providing a model backend [14]. Fortunately, many front-end tools have been designed to work with an Ollama backend, such as OpenWebUI and Chatbot Ollama [25],

[26]. However, no solution satisfied all our scaling needs, so having front-end expertise on the team was crucial for application modifications. Key features to consider include support for non-local backends, the ability to integrate with SSO, and the functionality to enable users to create custom retrieval augmented generation (RAG) systems [27], which also requires deploying a vector store like ChromaDB or PGVector. Figure 3 shows a sample deployment architecture. Finally, the pace of innovation means that while many frameworks are being released, there is not yet a consensus on the best approach to these tasks. Thus, building services modularly when possible and accounting for recurring costs to update the service to avoid technical debt as initial approaches become outdated are important. Support for RAG tools and agents for multi-hop reasoning has become very popular and should be considered a requirement for custom services in the long term.

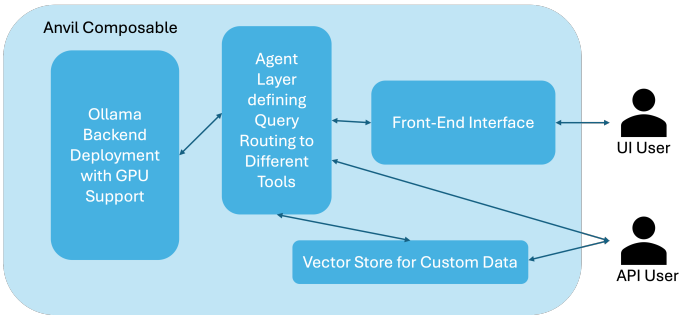


Fig. 3. Architecture of custom on-prem LLM service

III. Discussion

The ecosystem of GenAI tools evolves rapidly, and today's models and tools may differ significantly by the time Gateways '24 starts. Although our team at Purdue has thoroughly evaluated options for deploying a campus LLM, our specific results are less important than the broader insights gained. We believe other universities and research centers building similar gateways, and we hope our findings will guide their efforts. In this ever-changing environment, quickly adapting and assessing new developments is crucial for success.

IV. Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 2005632. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] Y. Liu et al., "Understanding LLMs: A Comprehensive Overview from Training to Inference," arXiv:2401.02038 [cs], Jan. 2024.

[2] T. Burns. "ITS debuts custom artificial intelligence services across U-M." University of Michigan. https://record.umich.edu/articles/its-debuts-customized-ai-services-to-u-m-community/ (accessed May 20, 2024).

[3] H. Crompton and D. Burke, "Artificial intelligence in higher education: The state of the field," Int J Educ Technol High Educ, vol. 20, no. 1, 22, Apr. 2023, doi:10.1186/s41239-023-00392-8.

[4] WhyLabs, "A Guide to Large Language Model Operations (LLMOps)," WhyLabs. https://whylabs.ai/blog/posts/guide-to-llmops (accessed May 26, 2024).

[5] OpenAI. "Data Controls FAQ." OpenAI. https://help.openai.com/en/articles/7730893-data-controls-faq (accessed May 20, 2024).

[6] OpenAI. "Introducing ChatGPT Enterprise." OpenAI. https://openai.com/blog/introducing-chatgpt-enterprise#OpenAI (accessed May 20, 2024).

[7] S. Heshmatisafa and M. Seppänen, "Exploring API-driven business models: Lessons learned from Amadeus's digital transformation." Digital Business, vol. 3, no. 1, 100055, Jan. 2023, doi:10.1016/j.digbus.2023.100055

[8] T. Hagendorff, "The Ethics of AI Ethics: An Evaluation of Guidelines," Minds and Machines, vol. 30, pp. 99-120, Feb. 2020, doi:10.1007/s11023-020-09517-8

[9] F. Kumeno, "Software engineering challenges for machine learning applications: A literature review," Intelligent Decision Technologies, vol. 13, no. 4, pp. 463-476, Feb 2020, doi:10.3233/IDT-190160

[10] X.C. Song et al. "Anvil - System Architecture and Experiences from Deployment and Early User Operations," in Practice and Experience in Advanced Research Computing (PEARC '22), 1–9.

[11] H. Touvron et al. "Llama 2: Open foundation and fine-tuned chat models," arXiv:2307.09288 [cs], July 2023.

[12] T. Brown et al., "Language Models are Few-Shot Learners," arXiv:2005.14165 [cs], May 2020.

[13] B. Jacob, et al. (2017). "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference," arXiv:1712.05877 [cs], Dec 2017.

[14] J. Morgan. "Ollama." Ollama. https://ollama.com (accessed Apr. 16, 2024).

[15] T. Shnitzer et al., "Large Language Model Routing with Benchmark Datasets." arXiv:2309.15789 [cs], Sept. 2023.

[16] J. Dodge, S. Gururangan, D. Card, R. Schwartz and N.A. Smith, "Show Your Work: Improved Reporting of Experimental Results," arXiv:1909.03004 [cs], Sept. 2019.

[17] "LLMPerf: Large Language Model Performance Benchmarking," GitHub repository, Ray Project. [Online]. Available: https://github.com/ray-project/llmperf. (accessed Feb 4, 2024).

[18] J. Thiyagalingam, M. Shankar, G. Fox, and T. Hey, "Scientific Machine Learning Benchmarks." Nature Reviews Physics, vol. 4, pp. 413-420, 2022, doi:10.1038/s42254-022-00441-7.

[19] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ Questions for Machine Comprehension of Text," arXiv:1606.05250 [cs], June 2016.

[20] X. Zhou et al, "Benchmarking microservice systems for software engineering research." In ICSE '18 Companion (pp. 323-324), doi: 10.1145/3183440.3194991.

[21] "LiteLLM," GitHub repository, Berri AI. [Online]. Available: https://github.com/BerriAI/litellm (accessed Mar 5, 2024).

[22] K. Senjab, S. Abbas, N. Ahmed, A.u.R. Khan, "A survey of Kubernetes scheduling algorithms," Journal of Cloud Computing: Advances, Systems and Applications, vol. 12, no. 1, 87, June 2023, doi:10.1186/s13677-023-00471-1.

[23] Z. Sun, and A.V. Miceli-Barone, "Scaling Behavior of Machine Translation with Large Language Models under Prompt Injection Attacks," in Proceedings of the First edition of the Workshop on the Scaling Behavior of Large Language Models (SCALE-LLM 2024), pp. 9-23, Mar. 2024, doi:https://doi.org/10.48550/arXiv.2403.09832.

[24] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," Communications of the ACM, vol. 51, no. 1, pp. 107-113, Jan 2008, doi:10.1145/1327452.1327492.

[25] T.J. Baek. "Open WebUI." Open WebUI. https://docs.openwebui.com (accessed Apr. 14, 2024).

[26] I. Fioravanti. "ChatBot Ollama." GitHub repository. https://github.com/ivanfioravanti/chatbot-ollama. (accessed Feb. 23, 2024).

[27] P. Lewis, et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in Proceedings of Advances in Neural Information Processing Systems: 33, pp. 9459-9474, Dec. 2020, doi:10.5555/3495724.3496517.

Community Growth: Achievements Enhanced by SGCI/SGX3 Services

Sandra Gesing
San Diego Supercomputer Center
US Research Software Engineer Association
Chicago, IL, USA
sgesing@ucsd.edu

Sean B. Cleveland
University of Hawaii-System
Honolulu, HI, USA
seanbc@hawaii.edu

Carol Song
Purdue University
West Lafayette, IN, USA
cxsong@purdue.edu

M. Drew LaMar
William & Mary
Williamsburg, VA, USA
mdlama@wm.edu

Claire Stirm
San Diego Supercomputer Center
La Jolla, California, USA
cstirm@ucsd.edu

***Abstract*—Community growth is one of the cornerstones contributing to the sustainability of a science gateway. Achieving community growth requires careful planning and a multifaceted approach. The Science Gateways Community Institute (SGCI) and the Center of Excellence for Science Gateways (SGX3) offer services such as UX advice, sustainability training via the Focus Week, and an annual conference to support the science gateway community with developers and users. This panel will discuss four successful use cases – QUBES, MyGeoHub, CHEESE, and the Hawaii Behavioral Health Dashboard – where the teams utilized various SGCI/SGX3 services, which significantly contributed to their community growth. The discussion will highlight specific strategies and outcomes from these use cases, providing valuable insights into the effective practices that drive community engagement and sustainability in science gateways. Additionally, panelists will share lessons learned and good practices that can be applied to other science gateways seeking to enhance their community presence and impact.**

***Index Terms*—SGX3, Center of Excellence, science gateways, community growth, SGCI**

I. Introduction

Sustainability is a significant concern for many science gateways. It depends on many aspects, such as funding cycles, the framework's maintainability, an evangelist's enthusiasm to sustain a science gateway, and community buy-in [1]. For many successful science gateways, there are also as many that have been not successful [2]. Sustaining a gateway requires careful planning, design, development, deployment, and maintenance, as well as attention to the needs and expectations of the target user community. The Science Gateways Community Institute (SGCI) [3] and the Center of Excellence for Science Gateways (SGX3) [4] aim to address topics such as sustainability and assist the community in various tasks, ranging from using to developing and providing science gateways. Established in 2016 under the National Science Foundation's (NSF) software sustainability institute program, SGCI's funding concluded in July 2023, though it continues to offer paid services to clients. SGX3, funded in August 2022, is part of the NSF Center of Excellence program. Both SGCI and SGX3 emphasize the importance of usability [5] [6], reusability, and sustainability in science gateways while focusing on community growth across all research domains.

Over the past six years, it has become apparent that SGCI's community growth has been more significant in computer science and engineering than in other research fields, highlighting the need for increased attention to the user community for continued growth. Consequently, SGX3 services have been restructured based on lessons learned from SGCI. These lessons are derived from various engagements with SGX3 activities, including demographics of conference attendees, client interactions, post-event surveys, and website metrics. Additionally, services have been reorganized to align with the NSF Center of Excellence's framework.

Below is the current list of SGX3 services:

1) Usability/User-experience (UX) evaluation and design engagements lasting up to three calendar months
2) Technology evaluation and gap analysis
3) Science Gateway architecture design
4) Once annually Science Gateways Focus Week sustainability sessions
5) Focus Week follow-on sustainability coaching
6) Summer Coding Institutes
7) Rising Stars program
8) Summer faculty and student internships
9) Science Gateway Hackathons
10) Gateways conference series
11) Gateways Central site for gateway listing, software listing, and partnership formation
12) SGX3 On the Road outreach to scientific communities where they meet
13) Blueprint Factory sessions to develop the future roadmap for Science Gateways serving new domain science needs

Community growth has been identified as a major aspect of

Identify applicable funding agency here. If none, delete this.

the sustainability of a science gateway. Active engagement and expansion of the user base elucidates the value of such science gateways as workspaces for sharing simulations, data and workflows that transform results into knowledge. Community growth fosters a diverse and dynamic ecosystem of users who contribute to developing, refining, and disseminating the gateway's resources. Collaborations contribute to creating new ideas, feedback, and expertise, which are crucial for maintaining the relevance and efficacy of the gateway.

Moreover, a thriving community enhances the gateway's resilience and adaptability. As more researchers adopt and integrate the gateway into their workflows, they create a robust support network that can collectively troubleshoot challenges and innovate solutions. This collaborative environment accelerates scientific progress and builds a sense of investment among users, encouraging sustained use and advocacy for the gateway. Fostering community growth is not merely a strategy for expanding user numbers but a fundamental necessity for ensuring a science gateway's long-term sustainability, relevance, and impact.

In this panel, we will discuss which of SGCI's and SGX3's services have contributed to the community growth in the examples of QUBES [7], MyGeoHub [8], CHEESE [9], and the Hawaii Behavioral Health Dashboard [10].

II. Background

SGCI was structured to provide a variety of services, training, and community engagement opportunities. SGCI structure:

- Incubator - training and consulting services with regard to financial and technological sustainability, cybersecurity, and usability/user-experience.
- Extended Developer Support (EDS) - placement of a 0.25 full-time equivalent technologist from SGCI staff into projects needing to solve difficult technical problems and instill best practices.
- Workforce Development - training developers in Science Gateway development skills, training faculty in incorporating Science Gateway development in curricula, student hackathons, and summer- and semester-long internship programs, all with a special focus on diversity.
- Scientific Software Collaborative - gathering of a comprehensive inventory of existing science gateways, software used in science gateways, and Tech Summit activities [1] that bring together Science Gateway stakeholders to develop software elements for the community.
- Community Engagement and Exchange - outwardly focusing on dissemination of SGCI results through an annual conference, a vibrant website, webinars, blogs, success stories, and a variety of other publications and postings for public dissemination.

SGX3 aims to extend computing resource access to diverse audiences with varying skill levels, expand the community of science gateway developers and users, and promote sustainable practices for science gateways. SGX3's main work thrusts include:

- Growing a Diverse Community
- Developing the Workforce
- Serving as Community Experts
- Envisioning the Future

To deliver on these four thrust goals, the following activities will happen under SGX3:

- Blueprint Factories - collaborate to understand the cyberinfrastructure needs of research communities and national-scale providers.
- Workforce Development - build a supportive HPC/Gateways community for teaching faculty, and provide training and support; offer opportunities for students to enhance their HPC/science gateway skills through coding institutes, hackathons, and internships.
- Consulting - provide expert advice on project lifecycle sustainability, user interface and design improvements, and develop technical roadmaps for science gateway projects.
- Gateways Central - reimagine the science gateways catalog to serve as a central point for stakeholder interaction.
- Science Gateway Outreach - engage with science gateways and domain scientists/scholars where they are (SGX3 on the Road), collect community knowledge, celebrate community achievements through the annual Gateways conference, and share information via the Science Gateways Community website and mailing list.

SGCI's funded period ended in July 2023, but it will continue to offer services on a paid basis. SGX3, operating under SGCI, will be a funded activity, and community members who require more intensive engagements can access these services through SGCI on a paid basis.

III. Success Stories

The teams behind the following successful science gateways have experienced significant community growth, partly due to the activities and services provided by SGCI/SGX3.

A. QUBES

The Quantitative Undergraduate Biology Education and Synthesis (QUBES) gateway was originally funded by the National Science Foundation from 2014 to 2022 and is now sustained as a project of the BioQUEST Curriculum Consortium. Since the inception of SGCI in 2016, QUBES has utilized nearly all services, including hands-on consulting services (software development and usability evaluation), Gateways Focus Week training, inclusion in the Gateway Catalog, the Gateways conference series, and the Workforce Development program. QUBES participated in one of the original sustainability training workshops, the SGCI Bootcamp - now known as Focus Week, in April 2017, which was critical in helping to create a vision and mission that extended beyond the initial proposal and adopted a business model primed for sustainability. The usability design guidance and software

[1] https://sciencegateways.org/our-services/sgci-tech-summit

development work measurably increased the publication, adoption, and adaptation of open educational resources (OER). QUBES now hosts nearly 3,000 OER, some of which interface with computational and data tools hosted on the Gateway. However, one of the most valuable SGX3 resources is the annual Gateways conference, which has created a community of practice around building, hosting, and sustaining Gateways.

B. MyGeoHub and CHEESE Gateways

During the development phase of the MyGeoHub geospatial science gateway, the MyGeoHub team participated in a Focus Week workshop where they specifically looked into the sustainability of the gateway from more of a business perspective, i.e., identifying the unique value, potential strategies of targeting different user groups with specific capabilities, and business funding model and ways to stay vibrant in the longer run. With a clear vision and plan, we secured a large grant that funded the development of several key data-driven workflow solutions. These capabilities enabled the team to expand partnerships, create new services, and provide research opportunities for our RSEs. MyGeoHub has been in operation for nearly 10 years and supports approximately 11,000 users annually, ranging from K-12 students to multi-disciplinary scientists. It continues to evolve to support emerging technologies and host new projects. The team also used the usability consulting service successfully. The CHEESE project (Cyber Human Ecosystem of Engaged Security Education) sought to provide hands-on experience with common cybersecurity issues and mitigation strategies via a web-based demonstration platform. Most of the users were K-12 students and undergraduates, so usability was a key concern. The CHEESE team partnered with the SGCI usability team to improve the usability of the CHEESE website and demonstration applications. Six usability tests were carried out which identified ten areas of improvement in the visibility of actions, UI intuitiveness, and a lack of instant feedback. The usability team also provided a heuristic evaluation and a cognitive walkthrough to evaluate the user-friendliness of common workflows on the website. Three key areas of improvement were identified: consistency and standards in user interface elements, improving a sense of direction in multi-step workflows, and including vital information on the page. The CHEESE team implemented over 50% of these recommendations, which greatly benefited the usability of the CHEESE gateway.

C. Hawaii Behavioral Health Dashboard

The State of Hawaii Behavioral Health Dashboard (HBHD) [10], is an interactive online platform that provides data on substance use, mental health, and crisis care in Hawaii. The dashboard was developed as part of the Overdose Data to Action (OD2A) project funded by the Centers for Disease Control and Prevention (CDC). The OD2A project aims to expand public health surveillance and use data to drive prevention strategies for drug-related misuse and overdose morbidity and mortality [11]. During the development of the HBHD, the development team partnered with SGCI to leverage usability and user-experience consulting services. The SGCI usability team evaluated the initial beta release of the HBHD throughout a three month engagement. The usability team and the HBHD team engaged in 4 phases: Initial questions & background about the dashboard, Competitive analysis of similar dashboard, user-experience audit of the existing dashboard and user-centered design recommendations and mockups. The outcomes of the usability engagement were the redesign of some of the dashboard navigation, interfaces involved in query/filtering, and design templates for three common patterns within the dashboard. The usability team provided a summary report of their in-depth evaluation of the HBHD and mockups and templates in Figma that the HBHD team could use to refactor the dashboard. The HBHD team incorporated the results of the usability analysis into a final dashboard product with a friendly interactive user interface that has served important substance use and overdose information related to Hawaii for over 34,000+ visits since October of 2022.

IV. Summary

Sustainability of science gateways has many facets and a healthy growth of the community of a science gateway contributes to its sustainability. SGCI and SGX3 offer services such UX consulting and sustainability training that can positively influence the community growth. Each of the panelists representing one of the science gateways will give a short introduction to their science gateways and the services they received from SGCI and SGX3. The Co-PI of SGX3 will give a brief overview on the services. Afterwards, we will discuss questions in the panel such as

- How has UX consulting provided by SGCI/SGX3 impacted user engagement and satisfaction within your science gateway community?
- In what ways has sustainability consulting influenced the long-term viability and growth of your science gateway's user base?
- Can you share specific examples of how improvements in user experience (UX) have led to increased community participation and collaboration?
- How have the consulting services on financial and technological sustainability helped in attracting and retaining a diverse group of users for your science gateway?
- What experiences and feedback have you gathered from your community that highlight the importance of UX and sustainability consulting in their ongoing use of the science gateway?

Questions from the audience for the panelists will be prioritized. We expect a lively discussion about which factors lead to community growth, lessons learned, impact of SGCI and SGX3 services and good practices that can be applied to other science gateways.

Acknowledgment

We would like to acknowledge NSF OAC 1547611 (SGCI), NSF OAC 2231406 (SGX3), and NSF IUSE 1446258,

1446269, and 1446284 (QUBES).

REFERENCES

[1] S. Gesing, M. Zentner, J. Casavan, B. Hillery, M. Vorvoreanu, R. Heiland, S. Marru, M. Pierce, N. Mullinix, and N. Maron, "Science gateways incubator: Software sustainability meets community needs," in *2017 IEEE 13th International Conference on e-Science (e-Science)*, 2017, pp. 477–485.

[2] P. Calyam, N. Wilkins-Diehr, M. Miller, E. H. Brookes, R. Arora, A. Chourasia, D. M. Jennewein, V. Nandigam, M. Drew LaMar, S. B. Cleveland, G. Newman, S. Wang, I. Zaslavsky, M. A. Cianfrocco, K. Ellett, D. Tarboton, K. G. Jeffery, Z. Zhao, J. González-Aranda, M. J. Perri, G. Tucker, L. Candela, T. Kiss, and S. Gesing, "Measuring success for a future vision: Defining impact in science gateways/virtual research environments," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 19, p. e6099, 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.6099

[3] S. Gesing, N. Wilkins-Diehr, M. Dahan, K. Lawrence, M. Zentner, M. Pierce, L. Hayden, and S. Marru, "Science gateways: The long road to the birth of an institute," in *Proceedings of HICSS 2017*, 01 2017.

[4] *SGX3: Novel Concepts to Enhance Knowledge and Extend the Community Around Science Gateways*. Zenodo, Oct. 2023. [Online]. Available: https://doi.org/10.5281/zenodo.10034892

[5] P. Parsons, S. Gesing, C. Stirm, and M. Zentner, "Sgci incubator and its role in workforce development: Lessons learned from training, consultancy, and building a community of community-builders for science gateways," in *Practice and Experience in Advanced Research Computing*, 2020, pp. 491–494.

[6] P. Parsons, Y.-C. Chen, Y.-S. Ho, K. A. Groothuis, B. Dentler, C. Stirm, S. Gesing, and M. Zentner, "Common usability problems and solutions for science gateways," 2020.

[7] S. S. Donovan, C. D. Eaton, T. Gower, K. Jenkins, D. LaMar, D. Poli, B. Sheehy, and J. M. Wojdak, "Qubes: a community focused on supporting teaching and learning in quantitative biology," Jan 2018. [Online]. Available: https://qubeshub.org/publications/226/1

[8] R. Kalyanam, L. Zhao, C. Song, L. Biehl, D. Kearney, I. L. Kim, J. Shin, N. Villoria, and V. Merwade, "Mygeohub—a sustainable and evolving geospatial science gateway," *Future Gener. Comput. Syst.*, vol. 94, no. C, p. 820–832, may 2019. [Online]. Available: https://doi.org/10.1016/j.future.2018.02.005

[9] M. Lambert, R. Kalyanam, R. Kooper, and B. Yang, "Securing cheesehub: A cloud-based, containerized cybersecurity education platform," in *Practice and Experience in Advanced Research Computing*, ser. PEARC '21. New York, NY, USA: Association for Computing Machinery, 2021. [Online]. Available: https://doi.org/10.1145/3437359.3465584

[10] "State of hawaii behavioral health dashboard," https://bh808.hawaii.gov/.

[11] "Overdose data to action," https://www.cdc.gov/drugoverdose/od2a/index.html, Centers for Disease Control Prevention.

Geoweaver: A Science Gateway for Reproducible and Scalable Disease Prediction Research Using Deep Learning

Sai Vivek Vangaveti, Jyoshmitha Reddy Paturi, Ziheng Sun*
Department of Geography and Geoinformation Science, Center for Spatial Information Science and Systems
George Mason University
Fairfax, VA, USA
svangave@gmu.edu; jpaturi@gmu.edu; zsun@gmu.edu (*)

***Abstract*— The growing complexities of medical data require advanced analytical methods for accurately predicting diseases from symptoms and genetic information. Geoweaver, a web-based workflow management system, facilitates the automation of developing and deploying disease prediction models by leveraging genetic data and deep learning techniques. We developed an AI pipeline for health data processing, model training, prediction, and evaluation using Geoweaver. This system is user-friendly and supports the execution and management of Python scripts at every stage of the workflow. Geoweaver enables the reproduction of previous AI experiments with minimal effort, significantly reducing the time and effort required for workflow creation. This allows researchers to focus on technical details and other inquiries. Our results demonstrate the effectiveness of Geoweaver in enhancing workflow reusability and reproducibility in health AI research.**

Keywords*— *Geoweaver, Deep Learning, Workflow Management, Disease Prediction, Reproducibility

I. INTRODUCTION

The integration of machine learning techniques with healthcare data has demonstrated significant potential in enhancing medical diagnosis and treatment outcomes. The explosion of medical data from sources such as genetic sequencing [1], electronic health records (EHRs) [2], and wearable devices necessitates the use of advanced analytical tools like Geoweaver. These tools are essential for effectively extracting and analyzing meaningful information to support patient diagnosis and treatment.

Disease prediction plays a critical role in early detection, prognosis, and personalized treatment planning. Traditional statistical methods often fall short when handling the complexity and high dimensionality of modern medical data. Deep learning, a subset of machine learning, has emerged as a powerful alternative, capable of identifying intricate patterns within vast datasets. By incorporating genetic data, deep learning models can provide insights into an individual's susceptibility to certain diseases, potentially revealing risks that conventional methods might miss.

Ensuring reproducibility in research allows findings to be independently verified and validated, which is crucial for scientific integrity. Scalability, on the other hand, ensures that these machine learning models can be seamlessly integrated into clinical workflows, facilitating real-time decision-making in healthcare settings.

Geoweaver is a new format of gateway, a decentralized portal designed to address the challenges associated with developing, deploying, and executing complex machine learning workflows [3]-[5]. It offers a user-friendly interface and automates repetitive tasks, enabling researchers to concentrate on scientific discovery rather than technical details. It supports the creation of reproducible, reusable, and tangible workflows, making it easier to share and replicate experiments across different research teams and institutions.

This study aims to showcase the effectiveness of Geoweaver in replicating and automating disease prediction experiments using deep learning and genetic data. Researchers can build and refine predictive models with greater efficiency and accuracy. This platform facilitates the reusability of machine learning workflows, accelerating the development of innovative healthcare solutions and improving patient outcomes.

II. BACKGROUND

A. Motivation

The complexity of medical data is continually increasing due to the integration of various data sources, including EHRs, genomic data, and data from wearable devices. EHRs, for instance, provide comprehensive patient histories, including medical diagnoses, prescriptions, and laboratory results. However, managing and analyzing such voluminous and diverse datasets require sophisticated tools and methodologies [6]. Accurate disease prediction is a critical component of modern healthcare. Predictive models can identify patients at risk of developing specific conditions, enabling early intervention and tailored treatment plans [7].

Deep learning has emerged as a powerful tool for processing and analyzing complex medical data. Techniques such as convolutional neural networks (CNNs) [8] and long short-term memory (LSTM) networks [9] are particularly effective in handling the multi-modal and sequential nature of medical data. Furthermore, the incorporation of genetic data into predictive models provides a deeper understanding of

NASA ACCESS program (#80NSSC21M0028), National Science Foundation award EAR-1947875 and EAR-1947893, and OAC-2117834.

disease mechanisms and potential therapeutic targets [7].

B. Challenges in Disease Prediction Modeling

Ensuring model understandability and traceability is paramount for the adoption of predictive models in healthcare. Transparency in model development involves thorough documentation of algorithms, parameters, and the decision-making processes involved in generating predictions. Such transparency not only builds trust among clinicians but also facilitates the accurate application of models in clinical settings. Effective visualization tools and user interfaces further aid in making models more accessible and easier to trace, allowing users to interact with and explore model outputs intuitively.

Reproducibility is a fundamental aspect of predictive modeling in healthcare, ensuring that models produce consistent and reliable results across different datasets and environments. One of the main challenges to reproducibility is data variability, stemming from differences in data collection methods, preprocessing steps, and patient populations. Standardizing these processes is crucial to mitigate variability and enhance reproducibility. Regular validation and testing of models across diverse datasets further ensure that models maintain their performance and reliability in various settings

Improving productivity and efficiency in model development is critical for advancing healthcare AI. By creating reusable elements, researchers can reduce the time and effort required to build new models, allowing them to focus on innovation and improvement rather than starting from scratch for each project.

C. Introduction to Geoweaver

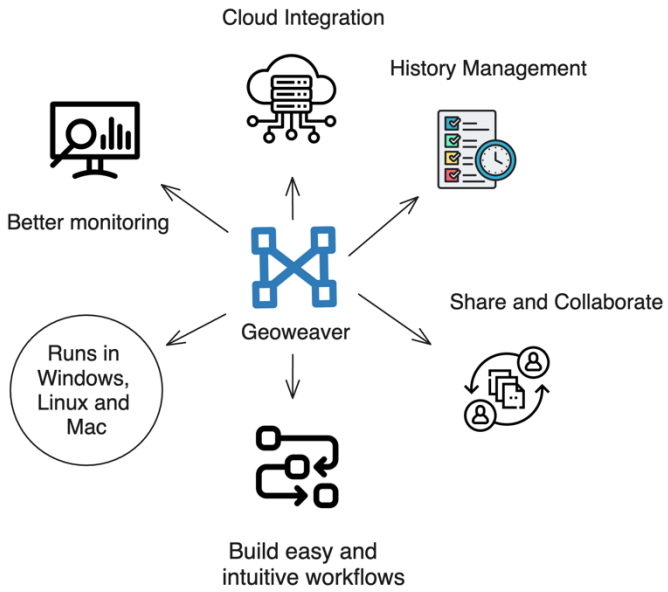


Fig. 1. Geoweaver features

Geoweaver is an open-source tool designed to manage and automate workflows in tangible, productive and less stressful way [10], [11], [12]. It provides a web-based interface that allows users to compose, execute, and share full-stack AI workflows seamlessly [13], [14]. Geoweaver supports Python and shell scripting, and it can run on Linux, Mac, and Windows platforms. One of its key features is the ability to record the history of each execution and change, ensuring reproducibility and traceability of AI experiments.

The platform integrates various processes, including shell scripts, and built-in processes, making it accessible for users with varying levels of programming expertise. Geoweaver also supports the orchestration of workflows across distributed environments, leveraging high-performance computation platforms and online spatial data facilities. Users can export and import workflows as package files, which include the source code and execution history, facilitating easy sharing and collaboration. Geoweaver is useful in building healthcare AI models into workflows. It streamlines the entire workflow management process, from data preprocessing to model deployment, and helps ensure that AI models are reproducible and reusable. A comprehensive comparison of Geoweaver's features with other workflow systems is in Table 1 of [15].

III. Experiment

A. Case Study: Disease Prediction

To demonstrate the reproducibility and reusability, we have chosen a disease prediction model using a decision tree from an open source github project [16]. The project explores the use of machine learning algorithms to predict diseases from symptoms, leveraging Naive Bayes, Decision Tree, Random Forest, and Gradient Boosting algorithms. Two datasets are utilized: one from Kaggle, containing 133 columns (132 symptoms and 1 prognosis), and another from Columbia University's Disease Symptom Knowledge Base, featuring columns for Disease, Count of Disease Occurrence, and Symptom. The project directory includes training and test data, pre-trained models, and scripts for loading, training, and saving models. The main.py script handles the Kaggle dataset, while a Jupyter notebook processes the Columbia dataset. Dependencies are managed via pip install -r requirements.txt. For demonstrations, an interactive demo can be run using Jupyter Notebook (demo.ipynb), and standalone inferences can be performed using infer.py. Note that this project is intended for demonstration purposes only and not for actual medical diagnosis.

We conducted the following steps to replicate the whole workflow in Geoweaver:

(1) Setting Up the Environment: The initial setup involves installing Geoweaver and configuring the host in Geoweaver and necessary environment. This includes using the requirements.yaml file, if provided in the project to install dependencies on the host machine. This file ensures that all required libraries and tools are correctly installed, streamlining the setup process.

(2) Importing the GitHub Project: The process begins by downloading the github repository to local machine, providing with a complete copy of all project files and necessary data to start developing on Geoweaver.

(3) Creating the Process: A process is a fundamental

component of a workflow. Each module in the project can be considered a process. A basic understanding of the project is necessary to import it into Geoweaver as processes. Python scripts can be imported as Python processes, while shell scripts are imported as shell processes. Step by step, add all the necessary modules for the project into Geoweaver. Modify the code as needed, such as adjusting file paths and dependency libraries, and then save the processes.

(4) Creating the Workflow: A workflow is created by integrating all the processes developed in the previous step, requiring a high-level overview of the model building to organize the processes into a coherent workflow for seamless execution. Geoweaver provides a workspace to create these workflows. Processes are added and arranged according to the model's architecture. Typically, the workflow follows a sequence of data collection, data preprocessing, model training, and model evaluation. Each relevant process is added to the workflow step by step. Once defined, the workflow is saved and executed in Geoweaver.

Geoweaver's monitoring interface allows for real-time tracking of the workflow execution, enabling users to promptly identify and address any issues. After each workflow run, a checkpoint is created, which can be used for future restoration. The workflow can be exported and shared along with the results.

The workflow is available for download as a zip file. This zip file contains a structured set of folders and essential files required for the workflow to function correctly. The primary files included in the zip file are:

workflow.json: Defines the overall structure and sequence of tasks in the workflow.

process.json: Contains details of each individual process

Code files necessary to execute the processes defined in the workflow.

B. Reproducibility and Reusability

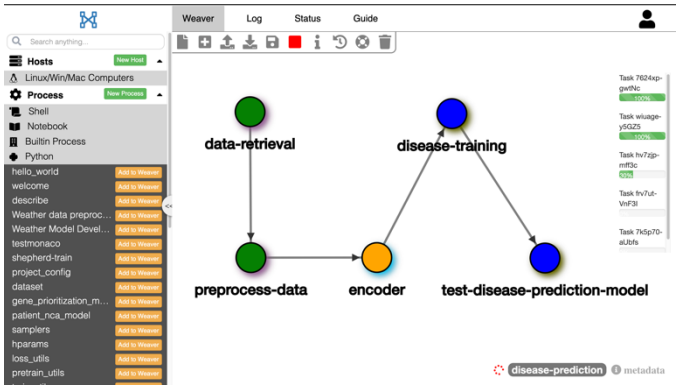


Figure 2. The final workflow in Geoweaver

Once a workflow is created, it can be shared among researchers, who can seamlessly import these workflows into their Geoweaver environments. This capability allows them to view past results and the entire history of the model from its inception to the current version. By providing the same data, researchers can reproduce the same results on their systems and continue working on the workflow.

Each workflow execution creates checkpoints that can be restored at any point, facilitating easy reproduction without the need to rewrite everything from the current version to any previous version. This feature is particularly useful when testing different models and making slight modifications to the processes, saving considerable time.

The disease prediction model created using this workflow is ready for deployment and can be shared with any Geoweaver user. They can not only view but also run and work on the model on their machines quickly and efficiently. Workflows generated from any version of GeoWeaver are fully compatible with all other versions of the platform. This compatibility ensures that results can be consistently replicated by uploading the workflows into any version of GeoWeaver. However, we plan to include tool versioning in future releases to enhance this functionality further.

Once a workflow is created, it can serve as a template for new workflows. Existing workflows can be modified to create entirely new workflows within Geoweaver, significantly reducing the effort and time required to build a model from scratch. This feature underscores the reusability of Geoweaver-created workflows, making the process of model development more efficient and streamlined.

C. Evaluation

Table 1. Qualification of the impacts of Geoweaver

Metric	Geoweaver	Without
Reproduce Rate	90%	85%
Reusability Rate	90%	80%
Flexibility	High	Low
Setup Time	2 hours	2 hours
Tangibility	90%	80%

The qualitative evaluation of using Geoweaver versus not using it for reproducing health AI work is based on our research estimates and team surveys (Table 1). Our research indicates that the reproduce rate with Geoweaver is 90%, compared to 85% without it, demonstrating a higher reliability in replicating results. Surveys conducted with our research team show that the reusability rate of workflows is 90% with Geoweaver, while it drops to 80% without it, highlighting Geoweaver's capability to efficiently reuse workflows and components. Geoweaver offers high flexibility according to our team, allowing for easy adaptation of workflows to new requirements, whereas the flexibility is rated low without the platform. Both setups require an average of 2 hours to configure, based on team feedback. The tangibility, or ease of understanding and interacting with the workflow, is rated at 90% with Geoweaver, compared to 80% without it. These metrics from our estimates and surveys underscore Geoweaver's effectiveness in enhancing the reproducibility, reusability, and overall efficiency of health AI research workflows.

IV. Discussion

A. Impact of Geoweaver on Disease Prediction Modeling

These workflows encapsulate the entire process of data preprocessing, model training, and model evaluation. Once

Identify applicable funding agency here. If none, delete this text box.

established, these workflows can be easily shared among researchers along with their results, allowing for collaborative improvements and consistent execution of the model. Geoweaver provides a comprehensive log management system. Outputs and logs of each process within the workflow are meticulously recorded and can be viewed under the respective process history in the log section. This feature ensures transparency and traceability, enabling researchers to track the execution flow and debug any issues efficiently. The availability of detailed logs for each step simplifies the process of validation and verification. Using these features of the Geoweaver researchers can build models effectively and quickly.

B. Limitation and Future Work

For small to medium-sized disease prediction models, reproducing them into workflows using Geoweaver is straightforward and efficient. However, when dealing with larger projects, the process can become time-consuming and complex. The increased number of processes and dependencies in large projects can lead to intricate workflows that require significant effort to manage and maintain. This complexity can pose challenges in ensuring all elements are correctly integrated and function seamlessly within the workflow.

Geoweaver does not support real-time collaboration yet, although workflows can be shared and exported for use by other researchers. This limitation can hinder the pace of teamwork and coordination, as researchers cannot collaborate on workflows simultaneously and must rely on asynchronous methods for sharing and integrating their work.

Ensuring all necessary dependencies and environments are correctly set up in the host system can be difficult, which might affect reproducibility and consistency of results. While Geoweaver provides a flexible environment for creating workflows, there may be limitations in integrating other tools required for highly specialized research.

V. Conclusion

This paper presents an in-depth analysis of reproducing a disease prediction model using Geoweaver, an advanced AI workflow management tool. The study focuses on evaluating the performance and efficiency of workflow reproduction with Geoweaver compared to traditional methods.

Acknowledgment

Thanks Mr. Vishesh Saluja for helping with pre-filtering health AI projects on GitHub.

References

[1] T. Starkweather, S. McDaniel, K. E. Mathias, L. D. Whitley, and C. Whitley, "A Comparison of Genetic Sequencing Operators.," in *ICGA*, 1991, pp. 69–76.

[2] R. S. Evans, "Electronic health records: then, now, and in the future," *Yearbook of medical informatics*, vol. 25, no. S 01, pp. S48–S61, 2016.

[3] "Geoweaver." Accessed: Jun. 04, 2024. [Online]. Available: geoweaver.dev

[4] "Geoweaver Online Demonstration Site." Nov. 27, 2023. [Online]. Available: https://geobrain.csiss.gmu.edu/Geoweaver/web/geoweaver

[5] Z. Sun, L. Di, A. Burgess, J. A. Tullis, and A. B. Magill, "Geoweaver: Advanced cyberinfrastructure for managing hybrid geoscientific AI workflows," *ISPRS International Journal of Geo-Information*, vol. 9, no. 2, p. 119, 2020.

[6] S. Dash, S. K. Shakyawar, M. Sharma, and S. Kaushik, "Big data in healthcare: management, analysis and future prospects," *Journal of big data*, vol. 6, no. 1, pp. 1–25, 2019.

[7] D. Zhang, C. Yin, J. Zeng, X. Yuan, and P. Zhang, "Combining structured and unstructured data for predictive models: a deep learning approach," *BMC medical informatics and decision making*, vol. 20, pp. 1–11, 2020.

[8] Z. Sun, L. Di, H. Fang, and A. Burgess, "Deep learning classification for crop types in north dakota," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 2200–2213, 2020.

[9] Z. Sun, L. Di, and H. Fang, "Using long short-term memory recurrent neural network in land cover classification on Landsat and Cropland data layer time series," *International journal of remote sensing*, vol. 40, no. 2, pp. 593–614, 2019.

[10] Z. Sun *et al.*, "Making Machine Learning-based Snow Water Equivalent Forecasting Research Productive and Reusable by Geoweaver," in *AGU Fall Meeting Abstracts*, 2022, pp. IN23A-04.

[11] D. Liljestrand, R. Johnson, K. Jain, J. Halgren, C. Oroza, and S. J. Burian, "Interactive Deep-Learning Modeling of Snow Water Equivalent Through Application of The National Snow Model With Geoweaver.," in *AGU Fall Meeting Abstracts*, 2022, pp. H32A-06.

[12] "Geoweaver: a web system to allow users to automatically record history and manage complicated scientific workflows in web browsers involving the online spatial data facilities, high-performance computation platforms, and open-source libraries." Accessed: Oct. 03, 2023. [Online]. Available: https://github.com/ESIPFed/Geoweaver

[13] A. Alnuaim, Z. Sun, and D. Islam, "AI for improving ozone forecasting," in *Artificial Intelligence in Earth Science*, Elsevier, 2023, pp. 247–269.

[14] N. C. Cristea, Z. Sun, A. A. Arendt, S. T. Henderson, M. Denolle, and A. Burgess, "GeoSMART: Machine Learning Training and Curriculum Development for Earth Science Studies," in *AGU Fall Meeting Abstracts*, 2022, pp. ED22B-0550.

[15] https://zenodo.org/records/10034694

[16] A. Dutt, "Disease Prediction based on Symptoms." Accessed: Jun. 04, 2024. [Online]. Available: https://github.com/anujdutt9/Disease-Prediction-from-Symptoms/tree/master

What is User Experience to Science Gateways?

Paul C. Parsons
Purdue University
West Lafayette, USA
parsonsp@purdue.edu

Soumya Krishnaraj
Purdue University
West Lafayette, USA
ssoumyak@purdue.edu

Max Berent-Spillson
Purdue University
West Lafayette, USA
mberents@purdue.edu

Jack Gerber
Purdue University
West Lafayette, USA
jogerber@purdue.edu

***Abstract*—User experience (UX) has become a central concept in fields involving human-technology interactions. However, despite its widespread recognition in other domains, UX has not been widely adopted in the science gateway community, where the narrower concept of usability remains the dominant focus. This paper explores the potential value of integrating UX concepts and methods into the design and evaluation of science gateways, emphasizing the distinction between usability and UX. Through an interview study conducted with gateways personnel, we investigate their perceptions of UX and its relevance to their work. Our findings suggest that while usability is considered important, UX offers additional benefits by encompassing affective, contextual, and experiential factors that usability alone may overlook. By incorporating UX concepts and methods, science gateways could improve user satisfaction and engagement, and also enhance the design and maintenance of gateway projects. We argue that a broader adoption of UX can enhance the design process, aligning gateways more closely with the diverse needs and experiences of their users.**

***Index Terms*—user experience, usability, science gateways, cyberinfrastructure**

I. Introduction

User experience (UX) is a core concept in fields that explore interactions between humans and technology, such as human-computer interaction (HCI), human factors engineering, and human-centered design. Despite decades of academic attention and its growing prominence in professional computing roles, UX has not seen widespread adoption within cyberinfrastructure research and science gateways. While the gateways community has increasingly focused on usability—a related but distinct concept—the broader UX perspective remains underexplored in this domain. Usability typically emphasizes performance metrics, such as how efficiently and effectively users can achieve specific goals within a system. UX, on the other hand, extends these concerns by incorporating affective, experiential, and contextual factors, offering new ways to think about user-system relationships.

User experience (UX) is a core concept in several fields dealing with interactions between humans and technology, including human-computer interaction, human factors engineering, and human-centered design. While the concept has been around for several decades in the academic literature [1], and has become increasingly popular as a professional role in computing fields [2], it has not seen the same prevalence in cyberinfrastructure and science gateways research and practice. The gateways community has seen increasing focus on usability [3], [4], which is a related but distinct concept. Compared to the performance focus that usability has, the concept of UX includes an expansion on the concepts and methods of usability, offering new ways to think about relationships between users and computing systems. In this paper, we present some findings from an interview study with gateways personnel with the goal of understanding their perceptions of UX and its value for gateways. We provide a brief outline of the origins and development of UX, and also comment on potential value of incorporating more UX concepts and methods into the design and evaluation of science gateways.

II. Background

The Human-Computer Interaction literature, where usability concepts and methods originated (e.g., [6]), has witnessed an expansion beyond simply the instrumental value of artifacts (i.e., their usability) to more focus on the *experience* of using them [5]. This expansion has encouraged thinking about topics like affect and meaning in addition to the traditional usability concerns of learnability, efficiency, and errors. In parallel to this expansion, there has also been an increasing emphasis on the *design* of artifacts rather than just their use. Traditional views of usability place little emphasis on the design process and methods that are employed to create software, instead focusing on the *evaluation* of software that is created [7].

Usability can be defined as "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use" [9]. Usability testing essentially tells us how well users can perform tasks that are given to them, and how confused, frustrated, or efficient they are. Focusing on usability alone still leaves many critical details about a product out—e.g., whether users actually want to use it, who the target user group is, whether it fits an important need, and how it should be designed. Other science communities have taken notice of the value of adopting a UX lens—e.g., the Pistoia alliance, a life sciences research organization, creating a model of UX heuristics to be used by other life science groups [8]. As evidenced by MacDonald et al. [10], UX practices have brought significant value to the field of libraries by revolutionizing how library services and resources are designed, delivered, and experienced by users. While usability is still a critical concept for gateways, adopting a broader set of UX concepts, methods, and practices could provide significant value for science gateways.

III. Method

We performed semi-structured interviews with gateways teams that previously worked with us through the Science Gateways Community Institute (SGCI) consulting services (see [11], [12]). We aimed for a diverse sample of participants from our population of 39 consulting engagements. When looking across previous engagements, we focused on 3 characteristics to achieve diversity: (1) the disciplinary context of the gateway; (2) the time of the consulting engagement; and (3) the maturity of the gateway at the time of consulting. We attempted to avoid selecting projects that were too similar across one or more of these dimensions. After mapping the projects across these dimensions, we identified a set of 25 projects that represented the best diversity from our population. All 25 teams were invited to participate in our study via email. Of the 25, 18 agreed to participate.

Once participants agreed to participate, they signed a study consent form and then filled out a brief survey. The survey collected basic information including the participant's role in the gateway, how long the participant has been involved in the gateway, the maturity of the gateway, the funding situation for the gateway, and the current software development support they have. Answers to these questions provided helpful information for the interviews.

Interviews were carried out from July 2021 through October 2021. Interviews were semi-structured and conducted remotely via videoconferencing. All interviews were recorded and then transcribed using an automated service called Otter.ai [14]. The interview protocol was developed based on our experiences with the consulting engagements, our understanding of the current literature, and our knowledge of cyberinfrastructure projects in general. The protocol consisted of three main topics: perceptions of UX, implementation barriers, and team factors. We started off by asking participants to provide a high-level overview of their gateway, and then what led them to seek out usability consulting services.

For the first topic, we asked questions about their knowledge and perception of the value of UX: what their knowledge of UX was prior to working with us, whether their knowledge changed as a result, whether their perceptions of the *value*, *role*, and *importance* of UX changed, and whether their future grant proposal strategies would change to account more for UX. For the second topic, we were primarily interested in how UX recommendations were implemented—in other words, what the "handoff" from us to the team looked like, and what challenges might exist in implementing UX recommendations. We asked participants to describe what happened during the consulting engagement, after they received final reports from the UX team, what their team discussions were like, how they made decisions about what to prioritize, and what barriers there were to taking action on recommendations that were given.

A. Participants

Eighteen participants were interviewed as part of this study. Table I provides details about each participant, including the disciplinary alignment of their gateway, their team size, gateway age, and when the consulting engagement took place. We achieved a relatively diverse sample, with gateways coming from several disciplines; a mix of smaller and larger team sizes; and a range of gateway maturity, including new gateways still in pre-release stage to gateways that have been in operation for more than 10 years; and engagements across the range of time in which we operated.

B. Analysis

Interviews lasted 55 minutes on average, with a range of 27 to 88 minutes. Collectively, the interviews were approximately 16 hours. All interviews were conducted by the lead author, and student researchers participated as note-takers across a range of the interviews. After transcription, transcripts were examined and errors and fixed manually by the research team. Once transcripts were corrected, they were loaded into a qualitative data analysis platform called Dovetail [15]. Transcripts were analyzed using hybrid thematic analysis [13], incorporating both bottom-up and top-down approaches.

IV. Findings

A. Understanding of UX

The majority of participants indicated a lack of familiarity with the concepts and practices of UX and usability. One participant described it as: "I think the concepts of UX are unfamiliar to a lot of people who are not kind of in the field", then, when continuing the discussion, used the two concepts as interchangeable: "I had learned a lot and that was my first exposure to this whole usability field." Another participant stated: "I had no idea before what it's [UX] for, and we learned a lot in terms of how people gather information on the interface is really critical to their learning process." Another participant described how PIs often do not have familiarity because they are focused on the scientific aspects of the work: "Most of the PIs on this have little or no experience doing this kind of user interface, or user focus work. They're really people who do, you know, do science."

Others described some awareness about what UX entails, although still at a superficial level. For example, a participant described their team's understanding of UX as: "we did not have an idea that it was super important beyond, you know, make it colorblind friendly, or whatever. But we weren't really aware of any design principles we should be following other than a vague idea of what we've seen on other websites, but not really knowing explicitly what those elements were." Others indicated a little more understanding, going from 'look and feel' to the importance of discoverability: "I think that we primarily focused on the look and feel. And then, you know, we're always hearing that it's difficult to find things on the website. And so it kind of like, we need to make it easier for people to find things. But we didn't really know how to do that, or what that really meant." Similarly, a different participant described it as "knowing our website didn't look great, but not really understanding what or how we could improve it."

TABLE I
GATEWAY CHARACTERISTICS FOR THE 18 PARTICIPANTS IN OUR STUDY: DISCIPLINARY ALIGNMENT OR CONTEXT OF GATEWAY; SIZE OF GATEWAY TEAM (PEOPLE); GATEWAY AGE (YEARS OF OPERATION); START DATE OF CONSULTING ENGAGEMENT. EACH ROW REPRESENTS ONE PARTICIPANT. PARTICIPANT IDS REMOVED TO MAINTAIN ANONYMITY

Disciplinary Alignment	**Team Size**	**Gateway Age**	**Consulting Date**
Oceanography/Geology	6-9	1-2	2021
Environmental/Ecological	6-9	6+	2021
Psychology/Genetics	3-5	<1	2020
Hydrology	10+	6+	2020
Environmental Science/Resource Management	3-5	6+	2020
Interdisciplinary Health	4-5	<1	2020
Microbiology/Data Science	10+	1-2	2020
Agriculture/Data Science	6-9	3-5	2020
Interdisciplinary Oceanography	1-2	3-5	2018
Hydrology/Geophysics	1-2	3-5	2019
Geoscience	3-5	3-5	2019
Ecology/Sociology	3-5	6+	2020
Ecology	1-2	1-2	2019
Ecology	3-5	6+	2020
Atmosphere Science	3-5	<1	2020
Wildlife Ecology	10+	6+	2020
Plant Biology	3-5	<1	2020
Resource Management	3-5	<1	2020

B. Value of UX

When asked about the value they put on UX and usability after working with our consulting team, one participant stated "absolutely, I can't overstate how important we feel usability is." Being shown the strategies and outcomes of UX work, this participant came to understand the value that usability and a design-oriented approach holds. Usability is not just a factor to consider, but an important concept that is essential to a platform's success. This user also mentioned focusing on usability-centered design going forward with their project, and their projects should be usability-focused from the start: "it [usability] has got to be centered in there from day one."

This leads to the important idea of where usability and UX fit into the planning and development process for science gateways, as exemplified by another participant: "At that point [during the design and development process], we know that we need to create some sort of version of this interface. And then we have to go and do some proper usability testing to understand, you know, does the manner in which information is being shown really, is that the way we want to go? Or do we make some big changes in how we're thinking about how our end users will interact with the interactive platform?" This highlights a fundamental aspect of UX that is often misunderstood or oversimplified within gateway development. While there is awareness of the importance of the user experience, this awareness frequently manifests at a later stage in the design process, after a gateway is already launched and problems are noticed. This approach exemplifies a reactive rather than proactive stance on UX, where it is considered a subsequent checkpoint rather than an integral, ongoing component of the design process. Another participant echoes this sentiment, stating how gateways folks "may have thought that this [UX] was kind of this add on." Such a viewpoint can treat UX as an optional enhancement—or even an afterthought—rather than a critical component of the development process.

V. DISCUSSION

A. UX Knowledge

Many participants commented on how little they knew about usability and UX before working with us. For participants that did have familiarity with these concepts, it was often limited to a focus on the 'look and feel' of product and the evaluation of it via usability testing. In other words, it was limited to usability, and even more limited on its evaluation late in the development process (rather than a proactive, design-oriented approach to user experience). Expanding conceptions of what it means to do user-centered kind of work can allow gateways creators to think more proactively about the value their platforms can bring to their community—e.g., thinking early on about who the target users are; what their goals and needs are; what opportunities and challenge exist in meeting those needs; what kinds of features might be useful, and how they should be organized coherently. These concerns all go beyond narrow conceptions of usability, and especially notions of considering usability only after a product is developed. We do not suggest that gateways PIs and developers need to become experts on UX; simply recognizing the broad range of concepts and methods—and the proactive, design-first approach—can help them find the right people to include in the planning and development of gateways projects.

B. Priorities

Several participants described having priorities other than those related to the user experience. For example, one of the participants described how their gateway has prioritized functionality ('does it work') over the user experience: "I would say we functioned a lot more on just does it work versus would the user be able to understand how to use it without detailed help guidance." This kind of approach exemplifies the so-called 'system-centered design' (vs. user-centered design) approach, where technical features and capabilities are

considered first, then the needs and preferences of users and other stakeholders are considered later. This approach is often accompanied by an expectation that users should read a manual. There are historical examples of systems with impressive technical capabilities that failed due to attending to user experience concerns too late or as a second priority (e.g., see Doug Engelbart's "Mother of all demos" that was eventually abandoned by ARPA).

C. Proactive vs Reactive Approach to UX

The insights gathered from our interviews indicate that while there is growing awareness of the importance of usability and user experience (UX) in science gateway development, there remains a need for a broader and more comprehensive understanding of UX. Expanding beyond usability testing, which is often conducted post-development, involves planning ahead and considering various use cases to ensure the user experience is an ongoing concern throughout the development cycle. Effective UX design requires incorporating problem framing and user research early in the process to gain a deep understanding of the target audience's needs and behaviors. After consulting with us, one participant recognized the importance of considering UX early, stating that it "seems like the most efficient thing, because I don't want to have to go back and fix problems. I would rather not build in the problems in the first place." This proactive, user-centered and design-oriented approach contributes to the sustainability of science gateways by ensuring that platforms are adaptable and resilient to changing requirements and technological advancements.

VI. Conclusion

Our study aims to shed light on the current understanding and perceptions of user experience (UX) among science gateway personnel. Despite the growing recognition of usability concerns within the community, there exists a significant gap in knowledge and awareness regarding the broader concepts and practices of UX. Many participants expressed limited familiarity with UX, highlighting the need for education and awareness-building initiatives within the science gateways community to leverage the full potential of UX in improving gateway design and functionality.

Our findings emphasize the critical importance of integrating UX considerations early in the development process rather than treating them as an afterthought. A proactive approach to UX, encompassing thorough user research, problem framing, and continuous feedback loops, is essential for creating effective and user-friendly science gateways. By prioritizing UX from the outset, teams can ensure that user needs are addressed comprehensively, leading to more sustainable projects.

We encourage science gateway personnel to recognize the value of UX practices and incorporate them into project planning stages. While they may not need to become UX experts themselves, understanding the fundamentals of UX enables them to make informed decisions about hiring the right professionals and allocating resources effectively. By fostering a culture of UX awareness and integration, science gateways can enhance their overall quality and impact, ultimately benefiting both users and stakeholders alike.

Acknowledgment

This work has been funded by NSF Awards ACI-1547611 and 2231406. Many thanks to our participants for volunteering their time.

References

[1] Lallemand, C., Gronier, G., and Koenig, V. (2015). User experience: A concept without consensus? Exploring practitioners' perspectives through an international survey. Computers in human behavior, 43, 35-48.

[2] Vorvoreanu, M., Gray, C. M., Parsons, P., and Rasche, N. (2017). Advancing UX education: A model for integrated studio pedagogy. In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (pp. 1441-1446).

[3] Lawrence, K. A., Zentner, M., Wilkins-Diehr, N., Wernert, J. A., Pierce, M., Marru, S., and Michael, S. (2015). Science gateways today and tomorrow: positive perspectives of nearly 5000 members of the research community. Concurrency and Computation: Practice and Experience, 27(16), 4252-4268.

[4] Parsons, P., Chen, Y. C., Ho, Y. S., Groothuis, K. A., Dentler, B., Stirm, C., ... and Zentner, M. (2020). Common usability problems and solutions for science gateways. Gateways 2020.

[5] McCarthy, J., and Wright, P. (2004). Technology as experience. interactions, 11(5), 42-43.

[6] Goodwin, N. C. (1987). Functionality and usability. Communications of the ACM, 30(3), 229-233.

[7] Fallman, D. (2003). Design-oriented human-computer interaction. In Proceedings of the SIGCHI conference on Human factors in computing systems (pp. 225-232).

[8] Pistoia Alliance User Experience for Life Sciences. https://www.pistoiaalliance.org/community/user-experience-for-life-sciences/. Accessed September 6, 2024.

[9] ISO 9241-210:2019 Ergonomics of human-system interaction — Part 210: Human-centred design for interactive systems. International Organization for Standardization, Geneva, Switzerland.

[10] MacDonald, C. M. (2017). "It takes a village": on UX librarianship and building UX capacity in libraries. Journal of Library Administration, 57(2), 194-214.

[11] Gesing, S., Zentner, M., Casavan, J., Hillery, B., Vorvoreanu, M., Heiland, R., ... and Maron, N. (2017). Science gateways incubator: Software sustainability meets community needs. In 2017 IEEE 13th International Conference on e-Science (e-Science) (pp. 477-485). IEEE.

[12] Parsons, P., Gesing, S., Stirm, C., and Zentner, M. (2020). SGCI Incubator and its Role in Workforce Development: Lessons Learned from Training, Consultancy, and Building a Community of Community-Builders for Science Gateways. In Practice and Experience in Advanced Research Computing (pp. 491-494).

[13] Fereday, J., and Muir-Cochrane, E. (2006). Demonstrating rigor using thematic analysis: A hybrid approach of inductive and deductive coding and theme development. International journal of qualitative methods, 5(1), 80-92.

[14] Otter.ai. https://otter.ai. Accessed September 6, 2024.

[15] Dovetail. https://dovetail.com. Accessed September 6, 2024.

HelioCloud Overview

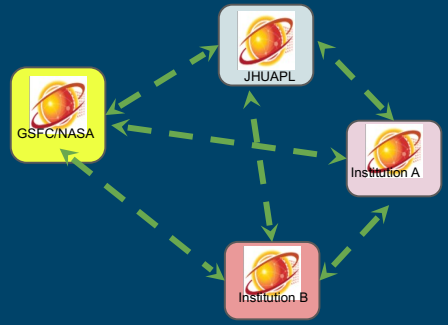
Shared Open Architecture

- **Instances are deployed at various institutions** and are interlinked (green arrows), sharing resources.
- **Open source**
- **Common software environments**
- **Data, service resources easily found & shared**
- **Browser-based Service Components**

High End Computing

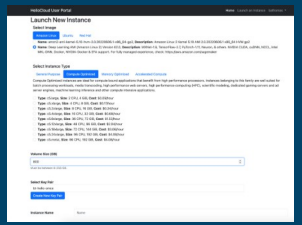
Research Collaboration

Open Science

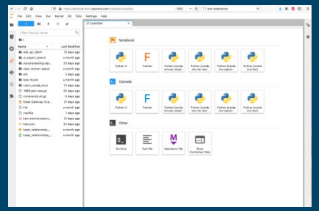


Network of HelioCloud Instances *deployed by institution which easily share resources. Currently GSFC, APL, TOPST-Helio, CU/Boulder (May 2024)*

Browser -based Service Components

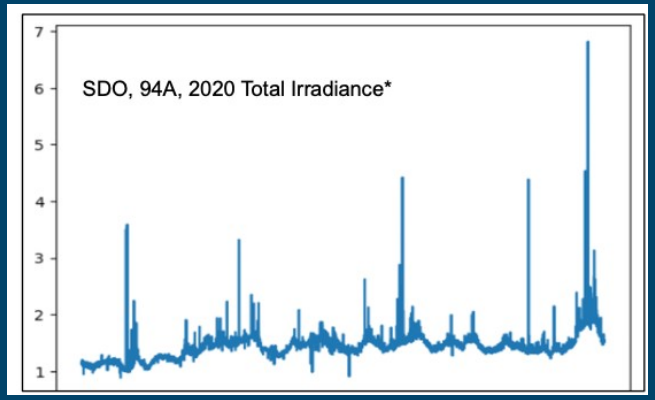


User Portal to create VMs in cloud (EC2)



Burstable Notebook environment which has GPU acceleration

Example: 2TB of SDO EUV data



Open source software & tutorials at *heliocloud.org*

Science Gateways Demo:
We will have a heliocloud instance stood up for a hands-on demo, and are happy to talk about the tech stack.

Science Use Case: One year of SDO 94A EUV images from AIA is 129,758 files, each 14MB, totalling 1.8 TB.

*A simple irradiance for 1 year. If done **on your laptop it would take 27 hours**.*

***HelioCloud takes 25 minutes** to analyze the same data*

About ~ 60x faster
Data I/O faster than 8Gb/sec!
Minimal code changes (python)

…and others can repeat the same thing!

Enhancing Science Gateways: Improving Access to HPC-ED Training Resources

SGX3 award # 2231406

Addressed Problem

Many institutions struggle to effectively interact with their training resources. Currently, HPC-ED relies on a command-line interface (CLI) for adding and querying training materials in its database. This approach is neither user-friendly nor intuitive and many potential users lack the necessary CLI skills. Consequently, they are unable to access and benefit from the institution's training resources.

Moreover, when seeking information, most people turn to Google, which often yields an overwhelming number of results. This makes it difficult to discern the quality and relevance of sources, leading to inefficiencies and potential misinformation.

Goals

1. **User-Friendly Interface**

Develop a web-based platform that simplifies the addition and querying of training materials without requiring CLI knowledge.

2. **Database Integration**

Connect to a database on HPC-ED to store, retrieve, update, and delete training resources (CRUD)

3. **Downloadable Resources**

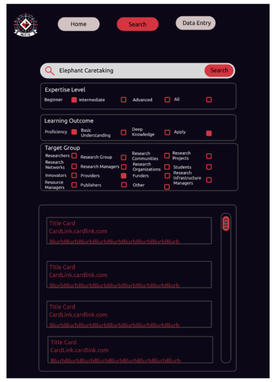
Implement functionality for users to download training materials in JSON format for offline access or further processing.

4. **Search Capabilities**

Incorporate search algorithms to ensure relevant and high-quality search results, minimizing the need to sift through numerous irrelevant entries.

5. **User Authentication/Permissions**

Set up a user authentication system to manage access levels, ensuring that only authorized users can add and modify resources.



Target Science Gateway

Our Targeted Science Gateway is the HPC-ED Gateway. HPC-ED (High-Performance Computing - Education) is a project to create and share metadata for HPC educational materials, making it easier to discover, access, and publish these resources through a federated catalog system.

HPC-ED
HPC-ED.GITHUB.IO

Resource Needs

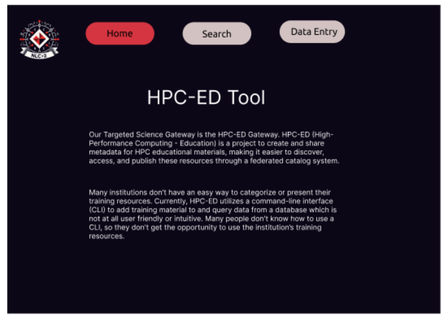
1. **Web development tools:** Front-end (HTML, CSS), Back-end Python(Django), Globus Search API
2. **Authentication Providers:** Globus, Google OAuth
3. **Hosting/Deployment:** GitHub, Docker
4. **User Interface:** Figma
5. **Collaboration Tools:** Slack, Zoom, Canva
6. **Documentation Platforms:** GitHub
7. **Team:** Frontend/Backend developers, UI/UX designers, documentation creators

Use Cases

Scenario 1: A graduate student new to HPC needs introductory resources to get started on her thesis project. She uses her university credentials via Globus Authentication to search for resources by inserting a text search.

Scenario 2: An institutional librarian wants to organize and make a collection of training resources available on HPC for students and faculty. She uses the platform to upload new training materials, categorize them using the tagging system to ensure they are searchable by relevant keywords.

Scenario 3: An IT staff Training Manager needs to provide his team with up-to-date resources on the latest HPC technologies. He uses the platform to find beginner and advanced training materials.



Methodology

Planning and Requirements Gathering
All team members participated in defining project goals and gathering user requirements. The team created a detailed project timeline and established key milestones.

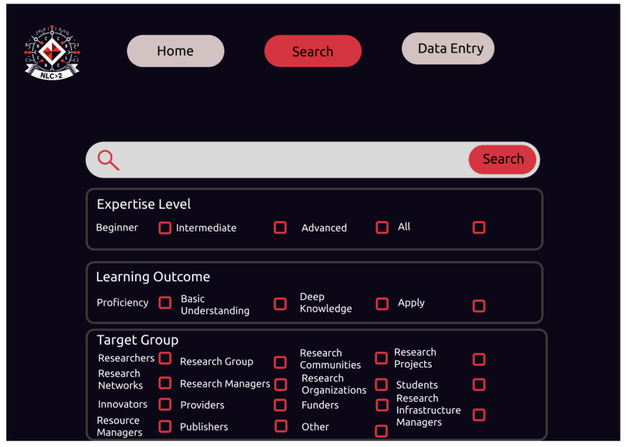
Design Phase
The design team created wireframes based on user requirements, focusing on ensuring a user-friendly experience.

Front-End Development
Front-end developers converted the wireframes into HTML and CSS, implementing interactive elements and ensuring intuitive navigation. They also integrated the front-end with back-end services using APIs.

Back-End Development
The back-end team set up the server environment using Python and Django. They implemented APIs for CRUD operations (Create, Read, Update, Delete) and implemented user authentication and authorization using Globus.

Integration and Testing
The team integrated the front-end with back-end APIs. They performed comprehensive integration testing and conducted usability testing with a group of potential users, identifying and fixing bugs and performance issues.

Deployment
The project is not deployed anywhere, only stored in GitHub. Future plans include containerizing the application with Docker and adding it to the public repository of Docker images.



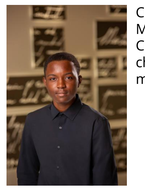
Possible Expansions

Given more time, there are several ideas and expansions we had in mind to further enhance this project. These include:

1. **Improved Search Capabilities:**
   Generate unique lists of filters based on the metadata in database entries, allowing users to refine their searches more effectively.
2. **Enhanced Authentication:**
   While authentication through Globus has been implemented, adding Google authentication would provide an additional layer of security, making the platform more secure and versatile for users.
3. **AI-Powered Resource Suggestions:**
   Integrate AI to provide resource suggestions based on user data and behavior. This would assist users in finding relevant training materials more efficiently, tailoring recommendations to their specific needs and interests.
4. **Responsive Design:**
   Ensure the platform is fully responsive and accessible on various devices, including desktops, tablets, and smartphones. This will cater to a wider range of users, providing a seamless experience regardless of the device being used.
5. **Community and Collaboration Features:**
   Add features such as discussion forums, resource sharing, and collaborative workspaces to foster a community of practice among users. These features will encourage collaboration, knowledge sharing, and peer support, enhancing the overall user experience.

By implementing these expansions, the platform can significantly improve the accessibility, usability, and effectiveness of training resources.

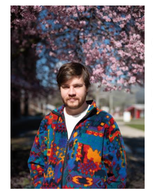
Authors



Christian Johnson
Morehouse College/SGX3
christianlj27@gmail.com

Lisha Ramon
SUNY Oneonta/SGX3
lisharamontn@gmail.com

Nole Stites
Southern Oregon University/
SGX3
nole.stites@gmail.com



Chandler Campbell
Southern Oregon University/SGX3
r.chandler.campbell@gmail.com

MORE INFORMATION → https://hackhpc.github.io/sgx3admi24

DATA DETECTIVES

FINDING SOLUTIONS TO EVERY PROBLEM!

TARGET SCIENCE GATEWAY

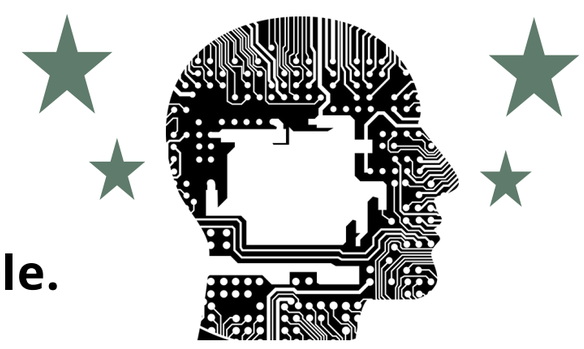
Our Targeted Science Gateway is the HPC-ED Gateway (High-Performance Computing - Education) which is an innovative project dedicated to enhancing the accessibility and dissemination of educational materials related to high-performance computing (HPC). The primary goal of this project is to create and share comprehensive information for HPC educational resources, thereby facilitating easier discovery, access, and publication of these valuable materials.

★ GOALS

- ***Goal 1:*** **Create flask database to digest information off websites**
- *Task 1: Make new project in Eureka/Jupiter*
- *Task 2: Develop flask app*
- *Task 3: Identify which websites to use*
- ***Goal 2:*** **Develop means to transfer information from sites to the app**
- *Task 1: Develop HTML file to connect to websites*
- *Task 2: Add validation to ensure data quality*
- *Task 3: Establish the form submission endpoint*
- ***Goal 3:*** **Transfer information from app to HPC-ED database**
- *Task 1: Ensure the data is easily user readable*
- *Task 2: Handle and store incoming form data*
- *Task 3: Transmit stored data to the HPC portal*

★ MISSION

- **The program is designed to gather, organize, and present HPC information in a way that enhances learning within the field. It collects content from various sources, ensuring a comprehensive range of materials. Once gathered, the program stores this information systematically, making it easily accessible and navigable. By presenting these materials clearly and concisely, the initiative aims to simplify complex HPC concepts and provide straightforward access to educational resources. This approach supports skill development and promotes effective use of computational techniques in both academic and professional settings, fostering a deeper understanding and practical application of HPC principles.**



IDENTIFIED ISSUES

HPC-ED wants to create a database to ensure an easier learning environment for HPC

RESOURCES

- **https://slack.com**
- **https://hpc-ed.github.io**
- **Grid Gain High Performance Computing**

SUPPORTERS

- **SGX3**
- **TACC**
- **SGCI**
- **Omnibond**

TECHNOLOGIES USED

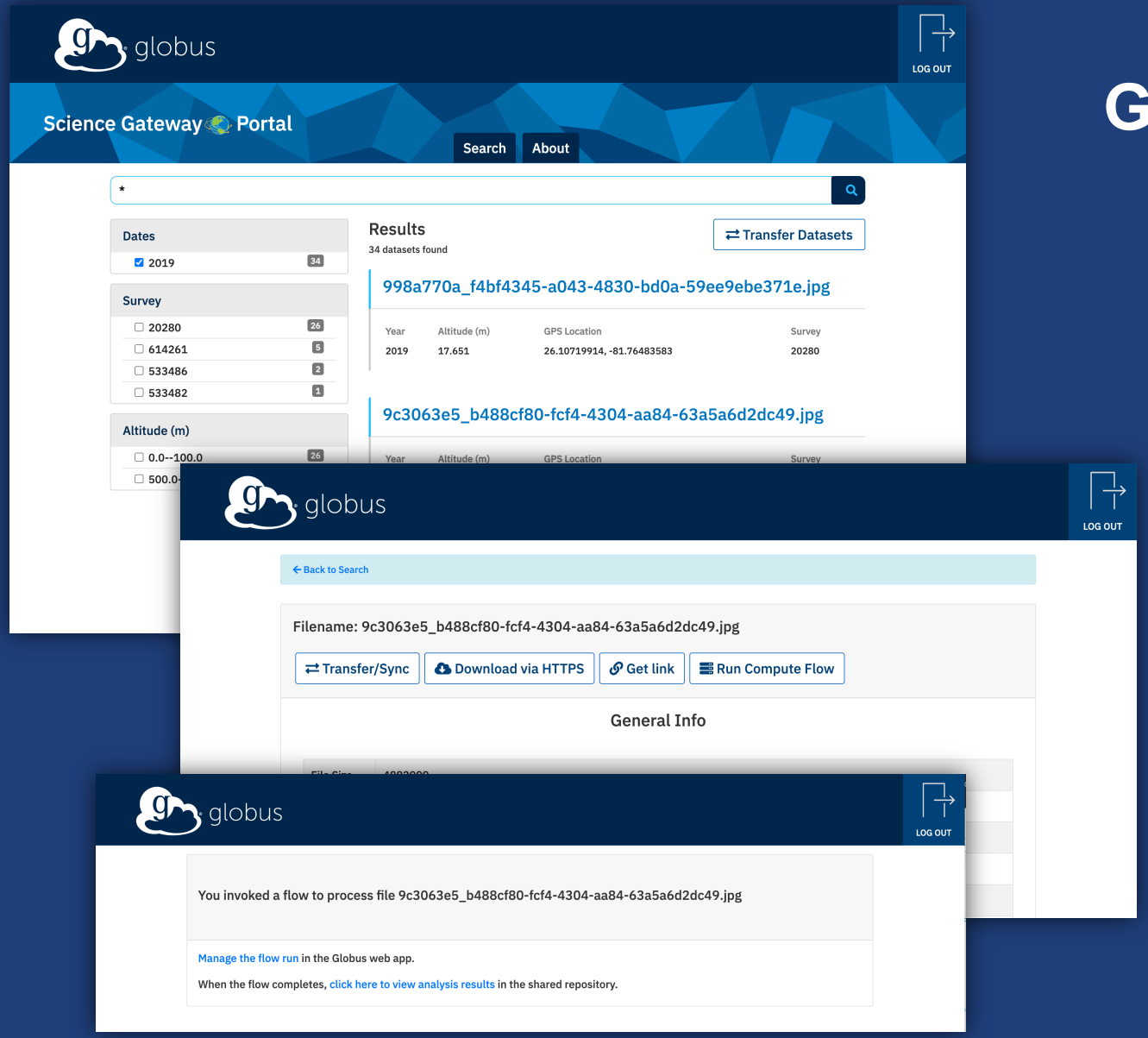
- **GitHub**
- **Python**
- **Requests, BeautifulSoup, Pandas**
- **Jupyter Labs**
- **Visual Studios**

Title: Wildland Fire Science Data Gateway

Authors: Kelley C. Barsanti, NSF National Center for Atmospheric Research; Samiha Binte-Shahid, University of California Riverside; Mona Wong-Barnum, San Diego Supercomputer Center

Keywords: wildfires, emissions, data, flexible querying, science gateway, drupal

Abstract: In the dry forests of the western US, long-term policies of wildfire suppression and past harvesting have led to the accumulation of understory fuels in many forests. This decades-long shift in forest structure, coupled with a warming climate, greatly increases the potential for destructive wildfires. Catastrophic wildfires across the western US and elsewhere have led to an increase in research aimed at elucidating linkages between fuels, combustion chemistry, fire emissions, plume chemistry, and plume rise with a goal of improving predictive smoke modeling. This has resulted in a large number of publications reporting relevant data, tools, and model advancements and applications. It is becoming increasingly difficult to maintain awareness of and make use of this rapidly expanding knowledge base. Here we present a prototype for a web-based resource that supports increasing the accessibility and utility of these resources. Examples of information that may be included are libraries of datasets and links to those datasets, software codes to facilitate data processing and use in model applications, links to relevant tutorials and trainings, and libraries of processed data and model inputs/outputs to facilitate use by air quality and smoke management communities. In the first phase of development, we have started with a collection of linked datasets providing emission factors (grams of pollutant/per kilogram of fuel burned) for globally-relevant fire and fuel types. The emission factors are stored in databases with varying levels of detail and data processing to allow flexible querying based on user needs. Our prototype Science Gateway facilitates data querying and data visualization, expanding the accessibility and utility of the data. It uses the Drupal framework hosted on the NSF-funded ACCESS cloud-based resource named Jetstream2. The current version enables user management, data management, and collaboration and includes a number of search features. Moving forward, we look forward to engagement with the Science Gateway community for further development, testing, marketing, and sustainability planning.



Globus Science Gateway Portal

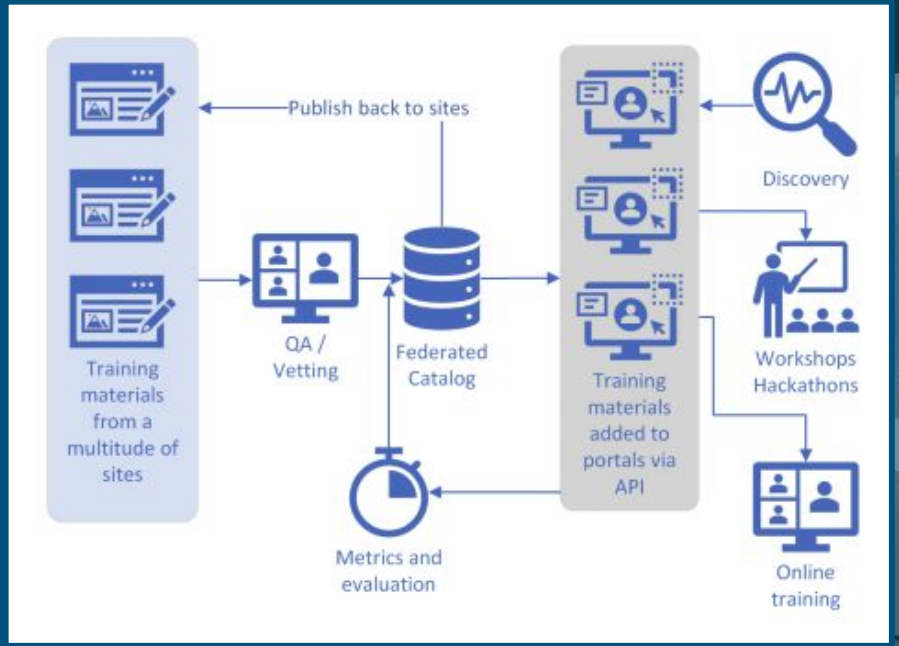
- **Browse** datasets via Globus Search
- **Discover**, select data of interest
- **Analyze** selected data using Globus Compute
- **Export**, **move**, **share** analysis results via Globus Transfer
- **Scale** with Globus Flows
- **Enable** broad access
  - Federated login and access control via Globus Auth
  - Extensible, Django-based framework
  - Template-driven search results and landing pages

https://github.com/globus/django-globus-portal-framework
https://sgportal.globusdemo.org

HPC ED

HPC-ED.GITHUB.IO

- **Project Goals:**
  - Make existing HPC Training and Education materials more findable
  - Create a federated catalog that can be easily searched
  - Facilitate community use of project tools and provide feedback



- Project site: https://hpc-ed.github.io
- Documentation: https://github.com/HPC-ED/HPC-ED.github.io/wiki

Building open education ecosystems to foster FAIRification between computational and data-centric tools and open educational resources

M. Drew LaMar
Department of Biology
William & Mary
Williamsburg, USA
https://orcid.org/0000-0002-4037-1848

Sam Donovan
BioQUEST Curriculum Consortium
Raymond, NH
https://orcid.org/0000-0003-4021-9878

Deborah L. Rook
BioQUEST Curriculum Consortium
Raymond, NH
https://orcid.org/0000-0002-3377-3638

The demand for, principles guiding, and productivity promises of open science have been well documented [1]. Still, awareness and adoption of FAIR standards (Findability, Accessibility, Interoperability, and Reusability) within research communities has been limited at best [2]–[4]. Recognizing common features across open science and open education, the United Nations Educational, Scientific, and Cultural Organization (UNESCO) has recommended that standards for open educational resources (OER) be embedded into policy frameworks that cut across open access, open data, and open source software [5]. SPARC [6] earlier described their "Open Agenda" in an effort to create a comprehensive platform for knowledge sharing and innovation by recognizing the centrality of OER.

While both open education and open science face challenges in making their resources accessible for reuse, it is the linkages across science and education that hold the greatest potential for innovation and broadening participation in science. The products of open science (which extend well beyond data) can be instrumental in the development of science skills in undergraduate STEM learning settings [7]. Additionally, the products of open education (which extend well beyond OER) can accelerate scientific discovery [8], [9]. Faculty must overcome barriers such as finding effective teaching resources, evaluating the appropriateness of datasets for use in their courses, and creating meaningful learning activities in unfamiliar computational environments. Like the limited uptake of FAIR and open practices in research communities, the adoption of open practices in education communities is still emerging and will require additional infrastructure support to ensure equitable implementation [10].

Open environments require the use of FAIR standards to operate at scale and maximize use by target communities [11]. Scientific gateways are community-developed cyberinfrastructure that use FAIR standards to describe open data, tools, and other resources using federated relational databases, and interoperability across platforms connecting users with open materials in the context of their professional work [12]–[14]. In order to build systems allowing for the robust integration of science and educational cyberinfrastructure there must be shared standards for describing, finding, accessing, and distributing open materials. Lowering barriers to resource use and developing metadata standards that facilitate access to educational and scientific products has the potential to drive innovation in STEM education, address challenges in the preparation of the technical workforce, and promote an informed citizenry that is empowered to engage in data-driven decision making.

In this poster, we will discuss the creation of a network of diverse stakeholders to establish, incubate, and sustain a set of Open Education Ecosystems (OEEs; see Fig. 1) designed to address the complex landscape of accessibility challenges faculty face when adopting computational and data-centric technologies for teaching and learning biology. Our network will be housed within the QUBES platform, a science education gateway [14] designed to lower barriers to faculty participation in STEM education reform. By working across the scientific and education communities and gateway cyberinfrastructures our network will leverage FAIR standards to streamline access to scientific data and tools for use in teaching and learning. We end with a call for participation from the gateways community.

National Science Foundation Award #2418329.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 2418329. QUBES would like to acknowledge NSF IUSE 1446258, 1446269, and 1446284, and the Hewlett Foundation.

References

[1] National Science Foundation, "NSF'S PUBLIC ACCESS PLAN: Today's Data, Tomorrow's Discoveries Increasing Access to the Results of Research Funded by the National Science Foundation," 2015, https://www.nsf.gov/pubs/2015/nsf15052/nsf15052.pdf.

[2] J. Brock, "A love letter to your future self: what scientists need to know about FAIR data," Nature Index, 2019, https://www.natureindex.com/news-blog/what-scientists-need-to-know-about-fair-data.

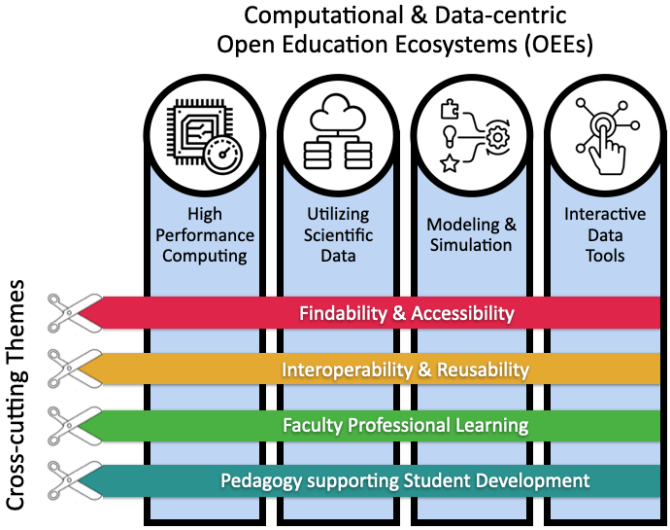


Fig. 1. Matrix of Open Education Ecosystems and Cross-cutting Themes. This figure provides examples of computational and data-centric use cases in education (vertical columns) around which OEEs will be established. A framework of shared cross-cutting themes highlights well established pinch points currently limiting the impacts of these tools in undergraduate biology education.

[3] B. Mons, E. Schultes, F. Liu, and A. Jacobsen, "The FAIR principles: First generation implementation choices and challenges," Data Intelligence, 2(1-2), 2020, pp.1–9.

[4] M.D. Wilkinson, M. Dumontier, I.J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.W. Boiten, L.B. da Silva Santos, P.E. Bourne, and J. Bouwman, "The FAIR Guiding Principles for scientific data management and stewardship," Scientific data, 3(1), 2016, pp.1–9.

[5] UNESCO, "Recommendation on open educational resources (OER)," Legal Instruments, 2019.

[6] H. Joseph, "We've made a few big changes at SPARC," 2016, https://sparcopen.org/news/2016/big-changes-at-sparc/.

[7] J.B. Labov, A.H. Reid, and K.R. Yamamoto, "Integrated biology and undergraduate science education: a new biology education for the twenty-first century?," CBE—Life Sciences Education, 9(1), 2010, pp.10–16.

[8] D.I. Hanauer, M.J. Graham, L. Betancur, A. Bobrownicki, S.G. Cresawn, R.A. Garlena, D. Jacobs-Sera, N. Kaufmann, W.H. Pope, and D.A. Russell, "An inclusive Research Education Community (iREC): Impact of the SEA-PHAGES program on research outcomes and student learning," Proceedings of the National Academy of Sciences, 114(51), 2017, pp.13531–13536.

[9] E.L. Dolan, "Course-based undergraduate research experiences: Current knowledge and future directions," Natl Res Counc Comm Pap, 1, 2016, pp.1–34.

[10] M.D. LaMar and S. Donovan, "FAIR-OS for Learning: Integrating FAIR principles and open practices to address shared societal challenges," Paper presented at the Mini-Gateways meeting (online) April 2022.

[11] L.D. Stein, "Towards a cyberinfrastructure for the biological sciences: progress, visions and challenges," Nature Reviews Genetics, 9(9), 2008, pp.678–688.

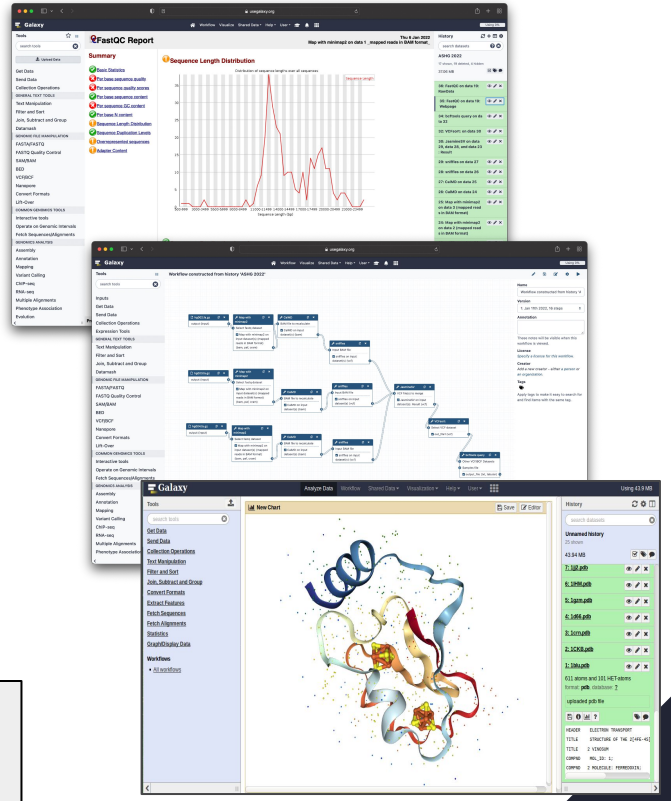
[12] N. Wilkins-Diehr, M. Zentner, M. Pierce, M. Dahan, K. Lawrence, L. Hayden, and N. Mullinix, "The science gateways community institute at two years," In Proceedings of the Practice and Experience on Advanced Research Computing, 2018, pp.1–8.

[13] P. Calyam, N. Wilkins-Diehr, M. Miller, E.H. Brookes, R. Arora, A. Chourasia, D.M. Jennewein, V. Nandigam, M. Drew LaMar, S.B. Cleveland, and G. Newman, "Measuring success for a future vision: Defining impact in science gateways/virtual research environments," Concurrency and Computation: Practice and Experience, 33(19), 2021, p.e6099.

[14] S. Donovan and M.D. LaMar, "Using Science Education Gateways to Improve Undergraduate STEM Education: The QUBES Platform as a Case Study," Computing in Science & Engineering, 25 (2), 2023, pp.20–29. doi: 10.1109/MCSE.2023.3292313.

What is Galaxy?

- An open platform for accessible, reproducible, and transparent computational biomedical research
- Toolshed with 1,000s of tools ready to run
- Terabytes of the latest, curated reference data
- Full featured workflow functionality
- Graphical interface for handling >1,000 samples
- Run Jupyter, RStudio, & Interactive Visualizations
- Extensive training tutorials and infrastructure
- Large international community of users and developers



All of this can be used on free public high performance infrastructure… or your institutional cluster… or the cloud… or your laptop… or a Raspberry Pi!

Galaxy PROJECT

Bridging Computational Science and Clinical Workflows with the ChRIS Research Integration System

Jennings Zhang, Rudolph Pienaar. — Boston Children's Hospital

Gateways2024 "Bring Your Own Portal (BYOP)" Submission

Name	ChRIS
URL	https://app.chrisproject.org
Description	The *ChRIS* Research Integration System is a platform for computational research and medical innovation. It provides a hub for collaboration on data analyses and a framework for creating medical applications. *ChRIS* itself is deployed on Kubernetes, while interfacing with legacy services in the hospital such as DICOM PACS.

Abstract

Despite the rise of AI and ML in medical imaging research, infrastructural challenges stifle the adoption of such technologies in clinical practice. To address this divide, we are developing ChRIS (a recursive acronym for the ChRIS Research Integration System). ChRIS facilitates the integration of computational research across various environments.

The IT of a typical hospital enterprise is a mixed bag: to accommodate big data, more and more research departments are moving their operations to public clouds such as AWS and GCP. However, existing solutions and legacy pipelines may only work on in-house high-performance computing (HPC) environments. Furthermore, clinical services tend to rely on outdated technologies such as the Digital Imaging and COmmunications in Medicine (DICOM) standard. The difference in IT between research and clinic in hospital settings exacerbates the lag in technological advancement on the clinical side. Thus, little of research innovation can go to directly impact patient care.

At its core, ChRIS is a platform for running computational workflows. It provides features for collaboration and data provenance. While the backend services of ChRIS run on Kubernetes to leverage cutting-edge features of the cloud-native ecosystem, ChRIS integrates with existing cyberinfrastructure and legacy services such as HPC schedulers and DICOM Picture Archival and Communication Systems (PACS). ChRIS is designed to support hybrid-cloud architectures to optimize the economic use of both on-premise and public cloud resource.

Within our research center, the PACS Query and Retrieve feature of ChRIS is the starting point of all of our neuroimaging research workflows. In the clinic, ChRIS is used to automate the execution of AI/ML computer vision algorithms on data from the PACS, granting radiologists access to cutting-edge tools for medical diagnosis. Across the world, ChRIS is used as the hub for scientific collaboration between institutions, including the HEALthy Brain and Child Development Study (HBCD). As MIT-licensed software, ChRIS is freely available for anyone to obtain and use.

ChRIS Research Integration System

ChRIS is a platform for medical compute.

ChRIS runs containerized pipelines *anywhere*:

- edge/on-prem
- HPC
- public cloud

... using any container engine.

podman

Red Hat OpenShift

HELM

APPTAINER

docker

kubernetes

ChRIS simplifies data retrieval from hospital databases (PACS).

CyberFaces: a scalable and flexible online platform for web-based learning and workforce development for advanced cyberinfrastructure

Instructors and students

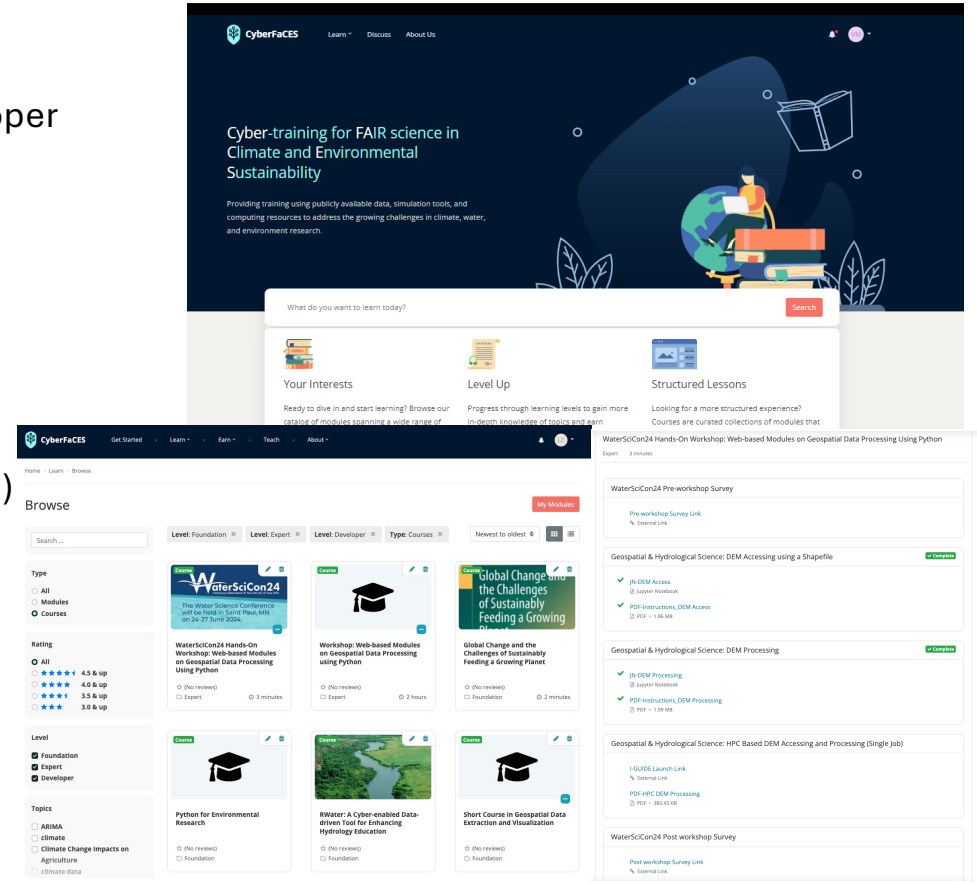
- Modular framework: Foundation, Expert, and Developer
- Curriculum delivery options
  - Modules
  - Courses
  - Badges and certificates
- Interactive learning materials (Jupyter Notebook)

System Design

- Open-source software stacks (Halcyon, JH, CILogon)
- Integration with GitHub, ACCESS
- Deployment on composable system
- Scalability support

Usage

- 85 modules and 11 courses
- Two in person workshops



Building a More Future Resilient Science Gateway

*

Matthew Potter
Johns Hopkins University
Applied Physics Laboratory

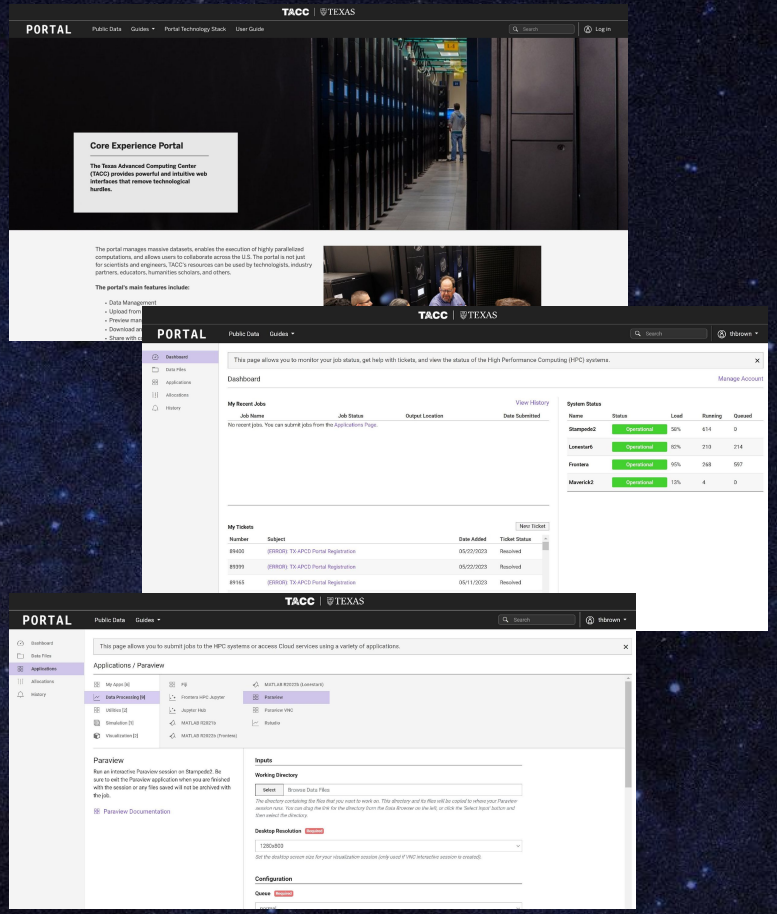
Laurel, Maryland
Matthew.Potter@jhuapl.edu

***Abstract*—Parker Solar Probe (PSP) nears the end of its nominal mission in 2025 and continues to produce ground-breaking scientific findings. In anticipation of an extended mission, the PSP Science Gateway was re-imagined for better maintainability, extensibility, and portability as a prototype for future missions. Originally built on a Drupal-based framework with heritage in the Van Allen Probes, the redesigned PSP science gateway relies on the REpresentation State Transfer (REST) approach for the backend architecture. The frontend user interface is managed using the React framework, providing an efficient and responsive user experience. User authorization and authentication is managed by a standalone server independent from both frontend and backend components, which provides simple but powerful role-based access for everyone, from the general public to privileged science team members. Many of the legacy HTML/JavaScript widgets were incompatible with the new React-based interface, so a simple IFRAME-based escape hatch technique was developed and integrated into the new generic science gateway UI framework. This architecture and many of the new capabilities of the PSP Science Gateway are presented and briefly discussed.**

TACC Core Experience Portal

- A robust open-source platform for building custom science gateways without starting from scratch
- Aggregates systems, services, APIs, and storage into a single, unified interface for researchers
- Offers a dashboard for real-time data storage, application execution, and job monitoring
- Integrates user storage with TACC's high-performance systems, simplifying data management
- Facilitates rapid deployment of new gateway projects with shared, customizable features

https://cep.tacc.utexas.edu | https://github.com/TACC/Core-Portal



Enhancing Tapis UI for ICICLE: Streamlined ETL Workflows and Dynamic Deployments

Introduction: Tapis is a multi-tenant, RESTful API framework designed for distributed computational research. It supports data management and code execution across institutional boundaries, enabling users to create portable, reproducible workflows. Tapis version 3, launched in September 2020, builds on NSF investments into the Abaco, Agave, and CHORDs projects. It supports over 80,000 users across 35 tenants and offers capabilities such as streaming/sensor data, containerized workflows, and a decentralized security kernel. Tapis UI is the user interface platform facilitating seamless interactions with Tapis services.

Enhancement of Tapis Framework for ICICLE: The enhancement of the Tapis Framework in support of the ICICLE project focuses on developing a user interface within Tapis UI to facilitate the creation of IKLE-specific Extract Transform Load (ETL) pipelines executed on Tapis Workflows. The work involves several key components and tasks to improve functionality, usability, and performance.

Primary Components and Tasks:

1. **ETL Branch for Tapis UI:**
    - Developed a specific deployment of Tapis UI in NGINX to support dynamic single/multi-tenant UI deployments based on configurations, ensuring flexibility and scalability for various user needs.
2. **Figma Planning:**
    - Conducted collaborative planning sessions using Figma to design a cohesive and user-friendly interface, ensuring that all elements of the UI are intuitive and efficient.
3. **DAG Library Implementation:**
    - Evaluated various Directed Acyclic Graph (DAG) libraries, such as d3, dag-builder-js, and reactflow, considering factors like

licensing, ease-of-use, and feature sets to choose the most appropriate library.

4. **UI as Configurations:**
    - Developed new deployer configurations for Tapis UI, allowing the customization of services, icons, and OAuth settings, enhancing the adaptability of the UI.
5. **ICICLE Tapis UI Extension NPM Package:**
    - Created an NPM package to extend Tapis UI, integrating OAuth configurations, Tapis client ID, base URL, and client key settings, streamlining the authentication process.
6. **Templated Tasks for DAG View:**
    - Implemented template tasks for IKLE ETL, enabling users to quickly add and edit nodes in the DAG view. This feature allows for rapid development and testing of ETL pipelines.
7. **Repository and Functions Files:**
    - Established a repository and created necessary function files to support the development and deployment of the new features.
8. **Sidebar with Templates:**
    - Developed a sidebar that allows users to click and add template nodes to the graph, with options for quick code edits and viewing stdout/stderr outputs, enhancing the user experience and simplifying workflow management.

Tapis System and Workflows: A Tapis system abstracts a host or cluster identified by name or IP address and is used for storing and retrieving files and data, running jobs (including staging files, executing jobs, and archiving results on remote systems). Tapis Workflows is an API and workflow engine that constructs and runs research computing workflows reproducibly within the Tapis ecosystem. The primary components of Tapis Workflows include:

- **Pipelines:** Collections of tasks performed during workflow execution.
- **Tasks:** Units of work, such as image builds, running arbitrary code, Tapis jobs, executing actors, sending HTTP requests, and running containerized applications.

- **Groups:** Collections of users that own workflow resources.
- **Archives:** Permanent storage for workflow results.

Significance of Tapis: Tapis offers programmable access to advanced resources, facilitating:

- Conducting analyses on cloud/high-throughput and HPC resources using a common API.
- Reproducing analyses with recorded inputs, outputs, and parameters.
- Sharing data, workflows, applications, and computational resources with collaborators, enabled by access controls within Tapis, without requiring the installation or support of a complicated technology stack.

The advancements in the Tapis Framework provide robust tools for the scientific community to conduct, reproduce, and share complex analyses efficiently.

Orchestrating End-to-End AI-Model Development using TAPIS and Smart Scheduler

1st Swathi Vallabhajosyula
Computer Science and Engineering
The Ohio State University
Columbus, Ohio, USA
vallabhajosyula.2@buckeyemail.osu.edu

2nd Nathan Freeman
Texas Advanced Computing Center
The University of Texas at Austin
Austin, Texas, USA
nfreeman@tacc.utexas.edu

3rd Christian Garcia
Texas Advanced Computing Center
The University of Texas at Austin
Austin, Texas, USA
cgarcia@tacc.utexas.edu

4th Joe Stubbs
Texas Advanced Computing Center
The University of Texas at Austin
Austin, Texas, USA
jstubbs@tacc.utexas.edu

5th Rajiv Ramnath
Computer Science and Engineering
The Ohio State University
Columbus, Ohio, USA
ramnath.6@osu.edu

ABSTRACT

With the increasing adoption of Machine Learning (ML) and Artificial Intelligence (AI), these technologies are being integrated into conventional sciences to analyze complex data, reveal patterns, and make precise predictions, thus enhancing research efficiency and discovery. Accurate AI model development requires capturing and labeling substantial data to create practical training datasets. Scientists must interact effectively with HPC systems like TACC[1] or OSC[2] to efficiently execute ML/AI workflows. Accurate resource estimation is vital for job scheduling and shared HPC resource management, yet users often face challenges that lead to job interruptions and inefficiencies. Despite existing guidelines, optimizing resource allocation for DNN workloads remains complex. AI-based resource estimation models, such as TPUGraphs [1] and Estimating GPUMemory [2], aim to address this but are difficult for end-users to utilize. Frameworks like TAPIS [3] and iScheduler [4] can streamline job scheduling on HPC systems. We have developed new TAPIS-UI components to facilitate efficient data labeling for conventional scientists, aiding in creating training datasets. This poster demonstrates an end-to-end use case by combining existing and new components to develop an AI-driven solution for detecting human intrusion using images captured by edge devices like camera traps. TAPIS, developed by TACC, offers APIs for managing scientific workflows, data, and computational resources. It automates tasks such as job submission, data transfer, and workflow orchestration across HPC resources and cloud environments. The iScheduler framework integrates AI-based resource estimators, validating predictions against a pre-cyberinfrastructure policy database before creating an execution plan. This plan is executed by an Intelligence Plane Server that handles job submission, monitoring, and rescheduling until completion.

[1]https://tacc.utexas.edu/
[2]https://www.osc.edu/

The poster outlines the following components for end-to-end AI model development using TAPIS and Smart Scheduler:

1) **Data Labeler:** A TAPIS-UI component that enables users to select and label datasets using an auto-labeling feature. The auto-labeler submits a pre-configured application based on the selected model as a TAPIS job and executes it on the user-chosen dataset to auto-label the images with animal, human, or vehicle tags. Once the auto-labeling job is executed, users can view and approve or adjust the labels using the Data Labeler to establish ground truth for training or fine-tuning an AI model.
2) **iScheduler Helper Notebook:** Provides APIs for interacting with the Smart Scheduler Framework, querying cyberinfrastructures like TACC and OSC, and facilitating job submissions based on system feasibility and wait time. The notebook accesses the pre-SLURM Cyberinfrastructure database, which contains information about available systems, configurations, and execution costs. It helps users estimate execution time and costs with various configurations and resources without submitting the jobs. This information allows users to manually submit a job or offload execution to the Smart Scheduler. The notebook can also submit jobs to the Smart Scheduler (a backend Intelligence Plane Server), which selects an appropriate system based on feasibility and wait time. The backend server uses TAPIS APIs to submit and monitor jobs, including callback hooks for tracking.
3) **View Smart Scheduler Jobs using TAPIS-UI:** Allows users to view and monitor jobs executed by the Smart Scheduler via TAPIS-UI.

Keywords: auto-labelling. generating training data, AI, ML, job scheduling, TAPIS, resource estimations, walltime predictions

ACKNOWLEDGMENT

This work was supported by the National Science Foundation's - AI Institute for Intelligent Cyberinfrastructure with Computational Learning in the Environment(ICICLE), Project Tapis: Next Generation Software for Distributed Research and SGX3 A Center of Excellence to Extend Access, Expand the Community, and Exemplify Good Practices for CI Through Science Gateways under grant agreements OAC-2112606, OAC-1931439 and OAC-2231406.

REFERENCES

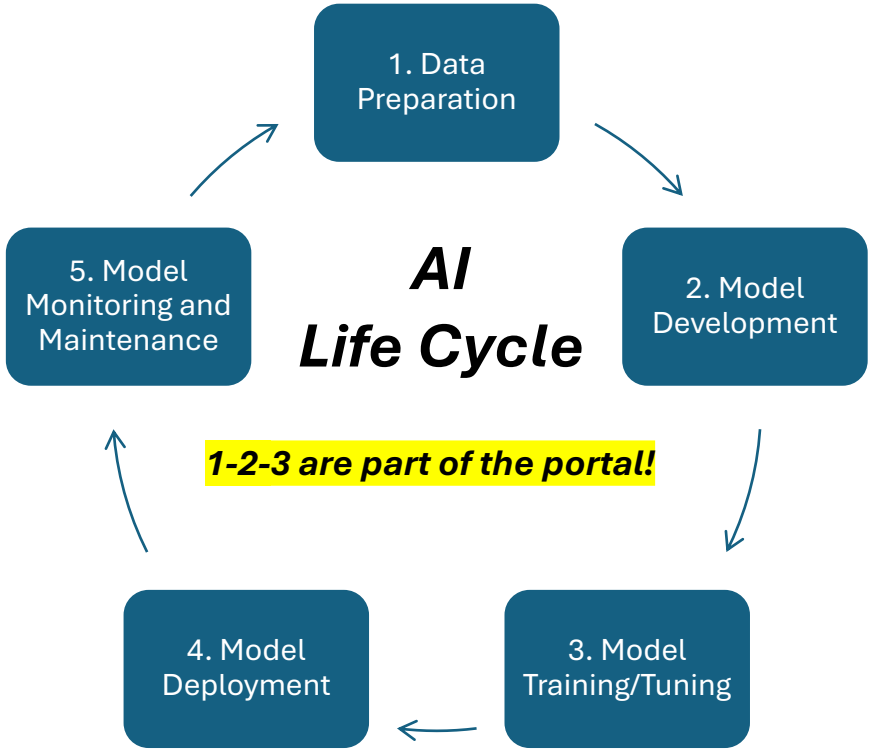
[1] Phothilimthana, M., Abu-El-Haija, S., Cao, K., Fatemi, B., Burrows, M., Mendis, C., & Perozzi, B. (2024). Tpugraphs: A performance prediction dataset on large tensor computational graphs. Advances in Neural Information Processing Systems, 36.

[2] Gao, Y., Liu, Y., Zhang, H., Li, Z., Zhu, Y., Lin, H., & Yang, M. (2020, November). Estimating GPU memory consumption of deep learning models. In Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (pp. 1342-1352).

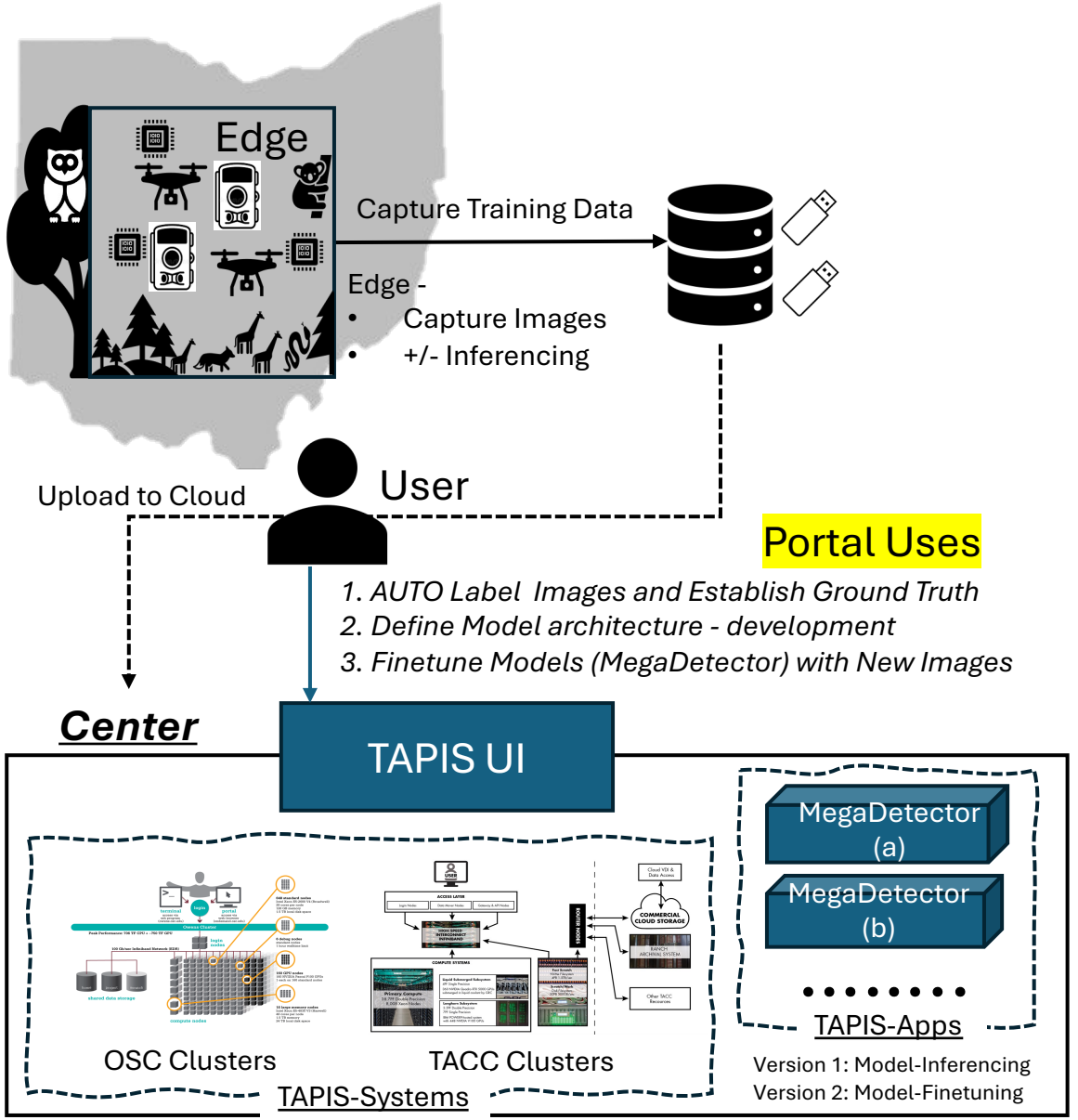
[3] Gao, Y., Liu, Y., Zhang, H., Li, Z., Zhu, Y., Lin, H., & Yang, M. (2020, November). Estimating GPU memory consumption of deep learning models. In Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (pp. 1342-1352).

[4] Vallabhajosyula, M. S., Budhya, S. S., & Ramnath, R. (2024). Reference Implementation of Smart Scheduler: A CI-Aware, AI-Driven Scheduling Framework for HPC Workloads. In Practice and Experience in Advanced Research Computing 2024: Human Powered Computing (pp. 1-4).

Data Labeling with TAPIS-UI: Establishing Ground Truth and Fine-Tuning the MegaDetector AI Model



One-stop solution for data preparation, model development, training, and deployment using TAPIS-UI

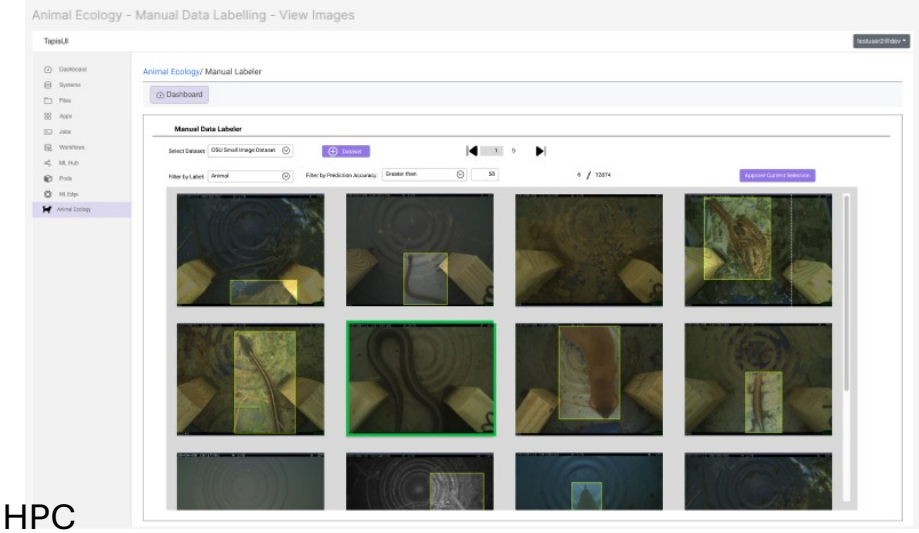


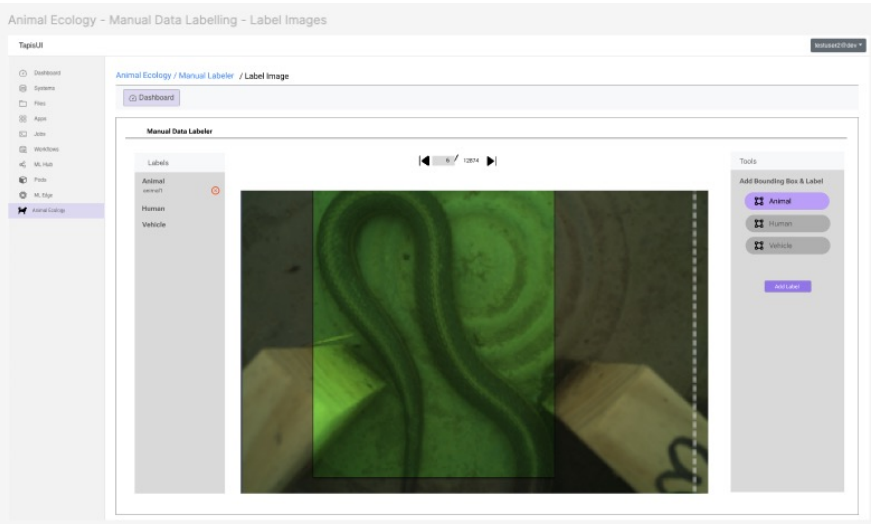
ICICLE DEMOCRATIZING AI

SGX3 | SGCI
A Center of Excellence and a Software Institute Serving the Science Gateways Community

Data Labeling with TAPIS-UI: Establishing Ground Truth and Fine-Tuning the MegaDetector AI Model

1. Label Images
- Auto-labeling using Model Inferencing (Using inference version of models registered as Apps (Version1))
- Human in the loop Ground Truth establishment after fine-tuning the auto-assigned labels (Using TAPIS UI).



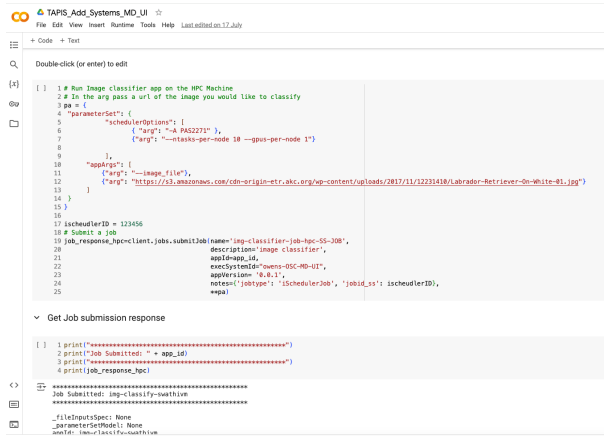


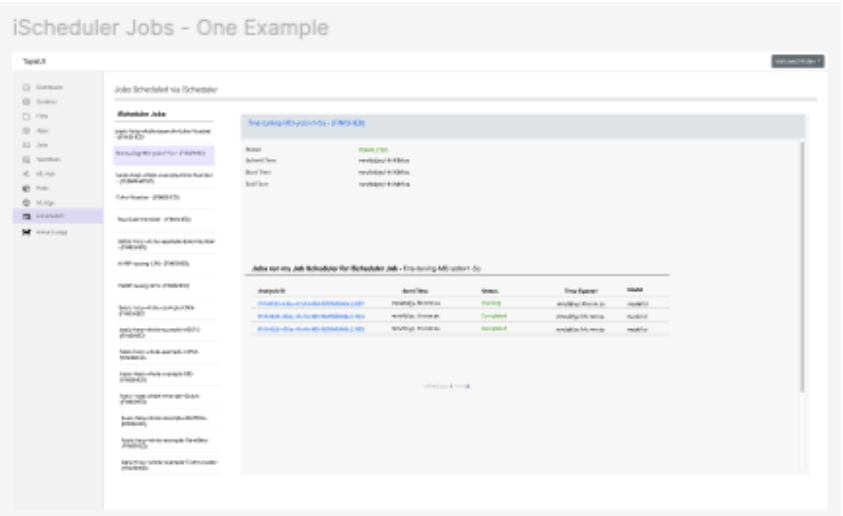
2. & 3 Auto-Schedule a fine-tuning HPC Job (Version 2 of Apps)

a. Submit Jobs for Fine-tuning – PODs Notebook – Smart Scheduler Interface

- APIs to interact with the Smart Scheduler Framework to fetch the resource needs and submit a job via the Intelligence Plane.
- Invoked during the model development phase to make informed decisions

b. View Using Tapis UI – Progress of training jobs.





ICICLE DEMOCRATIZING AI

SGX3 SGCI
A Center of Excellence and a Software Institute
Serving the Science Gateways Community

A Retrieval Augmented Generation Tool for Research Groups

Chandler Campbell
Department of Computer Science
Southern Oregon University
Ashland, OR
r.chandler.campbell@gmail.com

Bernie Boscoe
Department of Computer Science
Southern Oregon University
Ashland, OR
boscoeb@sou.edu

Tuan Do
Physics and Astronomy Department
University of California, Los Angeles
Los Angeles, CA
tdo@astro.ucla.edu

***Abstract*—Large language model (LLM) pipelines incorporating retrieval augmented generation (RAG) show great promise for indexing and querying large corpora of knowledge for scholarly research groups. However, the complexity and proprietary nature of most RAG-LLM applications presents a barrier to adoption for many research groups. In this poster, we present our ongoing work on AquiLLM, a web application built around a RAG-LLM pipeline that allows research groups to leverage the power of emerging artificial intelligence technologies to store and query knowledge. AquiLLM achieves this by augmenting existing commercial and open source LLMs with groups' knowledge bases. AquiLLM is independent of proprietary services and suitable for deployment on science gateways such as Jetstream2. AquiLLM uses a simple architecture, and incorporates mature, open-source technologies wherever possible to minimize risk and ease deployment. AquiLLM can be configured to give user groups complete custody of their data should they require it. We are currently rolling out AquiLLM to an astronomy research group as a proof-of-concept.**

Keywords—Retrieval Augmented Generation, LLM, science gateway, Jetstream2

I. Interface

AquiLLM can be deployed on a science gateway, and served to research group members as a website. AquiLLM presents two main workflows for users: ingestion and querying. AquiLLM can automatically ingest documents from arXiv and Zotero, allowing users to import and query existing knowledge bases. AquiLLM also supports ingesting LaTeX, PDF, ODF, Word, and raw text files manually. AquiLLM allows users to create collections of documents, and has a permissions model to restrict access to collections to specific users as desired by collection managers. Querying is through a familiar chat interface, similar to chatGPT and its competitors.

II. Architecture

At the core of most RAG pipelines is a vector database. Vector databases are used to store and retrieve latent space embeddings associated with pieces of text, where pieces of text which are close in meaning are near each other in the latent space. AquiLLM uses pgvector, an extension adding latent space embedding support to the decades old, open source PostgreSQL database, which many research groups will already be familiar with, and which can easily be deployed to Jetstream2 or another science gateway [1]. Latent space embeddings are generated using text embedding models. AquiLLM can be configured to use either commercial embedding models via an API, or open-source models run in a science gateway, depending on research groups' needs with respect to cost, as well as privacy considerations. AquiLLM manages the relationship between LLM and vector database internally, which avoids the complexity of integrating an external dependency such as Llamaindex or Langchain, and does so in an LLM-agnostic fashion [2]. This allows AquiLLM to be configured to use a variety of LLMs, both commercial and open source. Running an open source LLM in a science gateway allows users to keep custody of their data, making AquiLLM suitable for use with sensitive information. Every component of AquiLLM can be deployed on virtual machines in science gateways such as Jetstream2 [3].

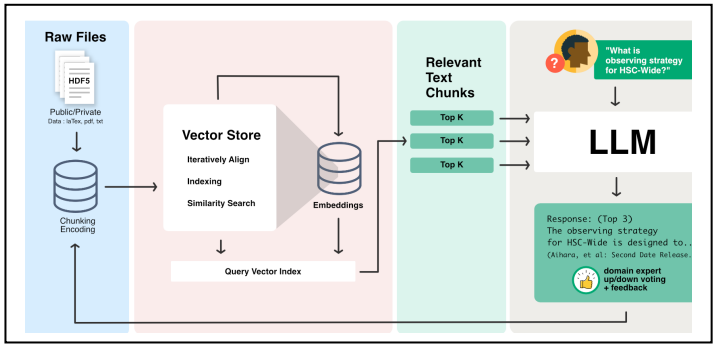


Figure 1: Workflow of RAG-LLM file processing and queries

References

[1] R. Aperdannier, M. Koeppel, T. Unger, S. Schacht, and S. K. Barkur, "Systematic Evaluation of Different Approaches on Embedding Search," in *Advances in Information and Communication*, K. Arai, Ed., Cham: Springer Nature Switzerland, 2024, pp. 526–536. doi: 10.1007/978-3-031-53963-3_36.

[2] W. Fan *et al.*, “A Survey on RAG Meeting LLMs: Towards Retrieval-Augmented Large Language Models,” Jun. 17, 2024, *arXiv*: arXiv:2405.06211. Accessed: Jul. 06, 2024. [Online]. Available: http://arxiv.org/abs/2405.06211

[3] D. Y. Hancock *et al.*, “Jetstream2: Accelerating cloud computing via Jetstream,” in *Practice and Experience in Advanced Research Computing*, Boston MA USA: ACM, Jul. 2021, pp. 1–8. doi: 10.1145/3437359.3465565.

Enhancing the Machine Learning Hub on Tapis: Designing and Deploying Models and Datasets Overview

Rahil Ashtari Mahini
Texas Advanced Computing Center
University of Texas at Austin
Rahil.AshtariMahini@austin.utexas.edu

Nathan Freeman
Texas Advanced Computing Center
University of Texas at Austin
nfreeman@tacc.utexas.edu

Dhanny Indrakusma
Texas Advanced Computing Center
University of Texas at Austin
dhannywi@utexas.edu

Gilbert Curbelo III
Texas Advanced Computing Center
University of Texas at Austin
gcurbelo@tacc.utexas.edu

Alexander Fields
Texas Advanced Computing Center
University of Texas at Austin
afields@tacc.utexas.edu

Joe Stubbs
Texas Advanced Computing Center
University of Texas at Austin
jstubbs@tacc.utexas.edu

Abstract The integration of machine learning (ML) into research has become increasingly essential for extracting valuable insights from complex datasets. However, the complexity of ML models can pose significant challenges for non-technical users. To address these issues, we have developed new features for the Machine Learning Hub (ML Hub) [1] in the Tapis UI [2], [3], aimed at simplifying ML model interaction and data management.

Working within the Texas Advanced Computing Center (TACC) and in collaboration with the Intelligent Cyberinfrastructure with Computational Learning in the Environment (ICICLE) AI Institute, we have developed and enhanced ML Hub to better support researchers and developers. Hugging Face's API [4] has been integrated into ML Hub, offering open-source pre-trained machine learning models. The Machine Learning Hub APIs [1], built with REST architecture and implemented in Python Flask, enables seamless access to these data.

We aim to develop a frontend to fetch and display models, and datasets' information from the integrated API. Machine learning hub also facilitates model training and model inference on the TACC Hyper performance computing (HPC) cluster. Currently, the ML Hub offers a comprehensive suite of features designed for non-technical users to easily access and utilize machine learning resources. These include:

1. **Models Overview and Download**: Users can explore and download a variety of machine learning models through a single, user-friendly interface. This includes model inference availability and detailed model cards.

2. **Datasets Overview and Download**: Similarly, users can browse and download datasets, with detailed dataset cards providing essential information.

These functionalities collectively provide a centralized access point for users to explore and download machine learning models and datasets, making advanced AI capabilities more accessible.

Our ongoing efforts include further enhancing the user experience by integrating training engines and inference clients, ensuring smooth operation within TACC's high-performance computing environment. By providing intuitive tools and interfaces, we aim to democratize access to advanced ML models and foster innovative research.

Keywords machine learning, Tapis, open-source, high-performance computing, user-friendly interface, science gateway, national science foundation.

Acknowledgements

This work was supported by the National Science Foundation's - Project Tapis: Next Generation Software for Distributed Research and SGX3 A Center of Excellence to Extend Access, Expand the Community, and Exemplify Good Practices for CI Through Science Gateways under grant agreements OAC-1931439 and OAC-2231406.

References

[1] Indrakusuma D, Freeman N, Stubbs J. Machine Learning Hub for Tapis; 2023.

[2] Stubbs J, Cardone R, Packard M, Jamthe A, Padhy S, Terry S, et al. Tapis: An API platform for reproducible, distributed computational research. In: Advances in Information and Communication: Proceedings of the 2021 Future of Information and Communication Conference (FICC), Volume 1. Springer; 2021. p. 878-900.

[3] Chuah JY, Rosenberg J, Strmiska K, Stubbs J, Cleveland S, McLean J. Tapis UI - A Rapid Deployment Serverless Science Gateway Built on the Tapis API; 2021. Accessed: July 29, 2024.

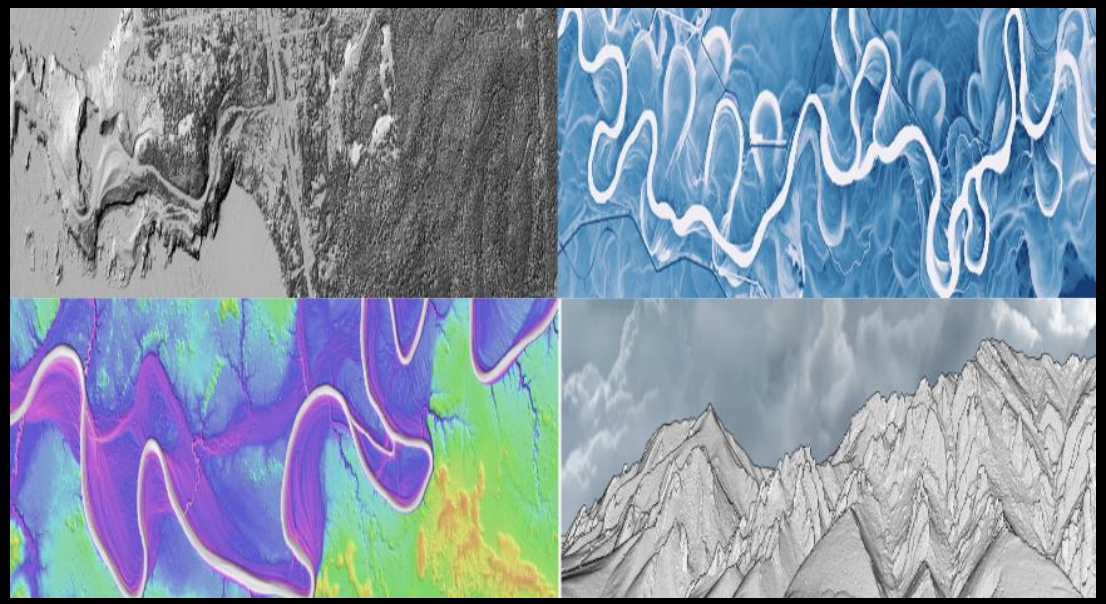
[4] Delangue C. Hugging Face: Hub client library; 2016. Accessed: July 29, 2024. Available from: `https://huggingface.co/docs/huggingface_hub/index`.

SGX3 | SGCI
A Center of Excellence and a Software Institute
Serving the Science Gateways Community

NSF NHERI DESIGNSAFE | Recon Portal

Extend Access, Expand the Community, and Exemplify Good Practices for CI through Science Gateways

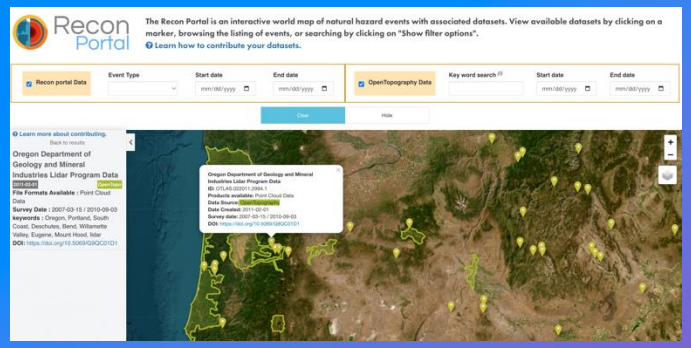
Integrating OpenTopography API into Design Safe Recon Portal for Enhanced Geospatial Data Analysis

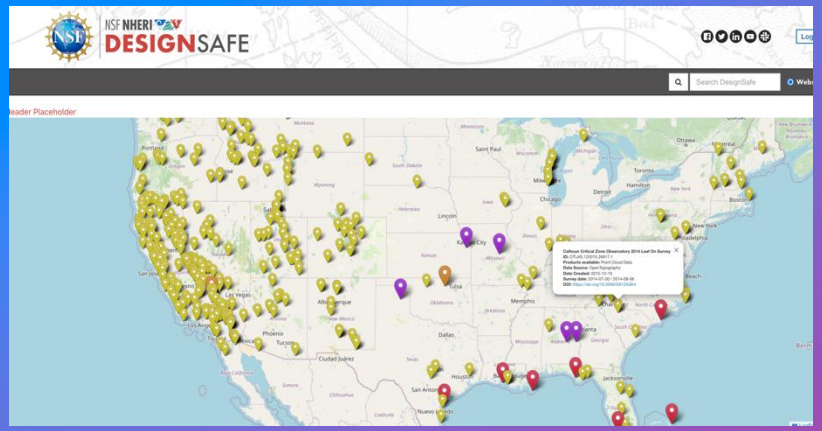


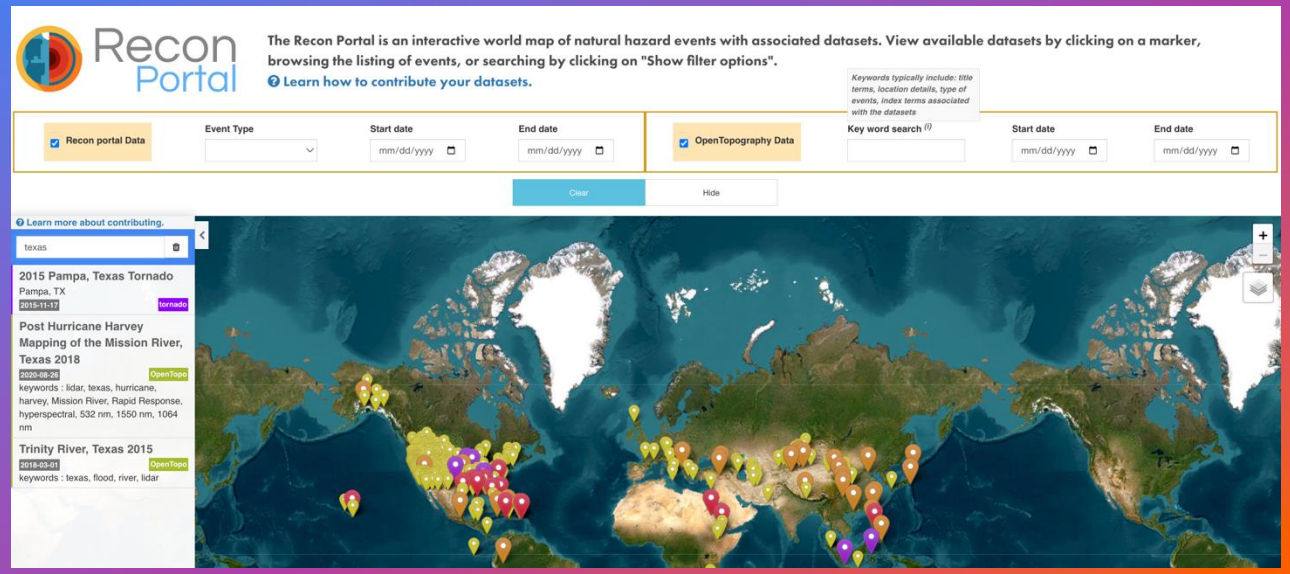
- **Web and Mobile Applications team at TACC**
  - **SGX3 Interns:** *Beulah Karrolla, Sajith Alapati*
  - **Mentors**: *Nathan Franklin, Frank Netscher*
  - **Manager:** *Tracy Brown*

Goal: *Collaborating to enhance the Design Safe Recon Portal's capabilities & Dedicated to advancing natural hazard research*

NSF awards
1547611
2231406







Exploring the integration of topographic data, including LIDAR point clouds and Digital Elevation Models, sourced from the OpenTopography API to enhance the portal's capabilities, providing a more comprehensive understanding of natural hazard events and their impacts.

Recon Portal Enhancements

- **OpenTopography Integration**: High-resolution topographic data for detailed spatial analysis and modeling
  - Center points data: Availability of topography data indicated on the world map
  - Multi polygons data: Detailed boundary definitions of geospatial areas
- **UI Enhancements**: Intuitive and visually informative map interface with color-coded event markers
- **Advanced Filtering**: Precise search refinement options for OpenTopography data
  - Author, title, keyword, description, location search, and date ranges
- **Secure File Retrieval**: Tapipy-enabled secure file fetching for data integrity and privacy
- **Modernized Web Portal**: Upgraded application framework for improved performance, scalability, and maintainability
  - Faster rendering and more responsive interactions
  - Efficient and modular codebase for developer

DesignSafe Recon Portal: Integrating OpenTopography Data for Enhanced Natural Hazard Analysis

Sajith Alapati, Beulah Karrolla
SGX3 Interns
Texas Advanced Computing Center, University of Texas at Austin
Austin, TX, USA
salapat@iu.edu, beulah.karrolla@austin.utexas.edu

***Abstract*—The DesignSafe Recon Portal is an interactive world map of natural hazard events grouped by their geographical locations, enabling researchers to log, view, and analyze data related to these events. This paper discusses the recent enhancements made to the portal, including the integration of topographic data such as LIDAR point clouds and Digital Elevation Models (DEMs) from OpenTopography. These enhancements provide researchers with comprehensive topographical data and context, offering new insights and improving the overall research capabilities. The transition from AngularJS to React with TypeScript further improves user experience and performance, enabling the addition of new features and interactive elements.**

***Index Terms*—DesignSafe, Recon Portal, OpenTopography, natural hazards, LIDAR, Digital Elevation Models, geospatial data**

I. Introduction

The DesignSafe Recon Portal serves as an interactive tool for researchers studying natural hazards. It groups natural hazard publications together by their geographical locations, allowing for detailed analysis and visualization. Recent enhancements have significantly improved the portal's capabilities by integrating additional geospatial data from OpenTopography [1]. This integration aims to provide comprehensive topographical data and context, assisting researchers in their analysis and understanding of natural hazards.

II. Enhancements and Features

The integration of OpenTopography data into the DesignSafe Recon Portal introduces several key features and improvements. Combining existing DesignSafe data with OpenTopography datasets offers detailed topographical insights into geographical regions. Researchers can use this data to determine how topography may have changed before and after natural disasters, but more importantly, it provides additional data and context to the geographic area.

Some of the notable enhancements include:

This work was supported by the National Science Foundation's Project Tapis: Next Generation Software for Distributed Research and SGX3 A Center of Excellence to Extend Access, Expand the Community, and Exemplify Good Practices for CI Through Science Gateways under grant agreements OAC-1931439 and OAC-2231406.

A. Interactive UI Enhancements

The portal now includes several interactive features to enhance user experience:

- *Polygon Mapping:* Users can view and interact with detailed polygon mappings of datasets, providing a visual representation of data points and their geographical extents. Polygons are turned on and off at certain map scales to improve clarity and performance [2].
- *Color-Coded Markers:* Markers on the map are color-coded based on the type of event and data source, making it easier to identify and differentiate between various events at a glance.
- *Advanced Filters:* New filtering capabilities allow researchers to narrow down their search based on specific criteria, enhancing the efficiency and effectiveness of data navigation and exploration.

B. Improved User Experience

The transition from legacy AngularJS to a modern React with TypeScript framework has several advantages [3] [4]:

- *Enhanced Performance:* The new framework ensures a dynamic and up-to-date user interface that leverages the latest technologies, providing a more seamless and responsive user experience.
- *Improved Testing and Maintenance:* The adoption of TypeScript improves the ease of testing and maintaining the codebase, ensuring long-term reliability and extensibility.

III. Implementation Details

The integration of OpenTopography data and migration to React with TypeScript involved several key steps.

A. Data Integration

Extensive research on OpenTopography APIs identified the Geoserver API as the best option for accessing comprehensive metadata and polygon coordinates. Scripts were developed to fetch, preprocess, and integrate this data into the portal's backend using Django. This included automating data fetching

and preprocessing at regular intervals to ensure up-to-date information [5].

B. UI Enhancements

The transition to React with TypeScript facilitated the implementation of several new interactive features:

- **Polygon Mapping and Marker Implementation:** Detailed polygon mappings visualize geographical data extents, enabled at specific map scales for clarity. Color-coded markers based on event types and data sources enhance visual differentiation.
- **Advanced Filters and Layer Controls:** Advanced filtering capabilities allow researchers to toggle data visibility on the map based on data sources and event types. Enhanced user interactions through a responsive UI dynamically update based on filter inputs.

C. Backend Optimization

Significant backend optimizations supported the new data integration and UI features. Implementing a caching solution using Django's cache framework with Memcached reduced data fetching times, improving overall performance.

D. Migration to React with TypeScript

Migrating the portal from AngularJS to React with TypeScript involved defining TanStack queries for data fetching and state management, developing base components to render the Leaflet map with new data integrations, and ensuring feature parity with the existing portal while leveraging React's capabilities for future extensibility [2] [3] [4].

IV. Challenges and Solutions

Several challenges were encountered during the integration:

A. API Limitations

Initial APIs lacked necessary data, requiring multiple integrations and feedback loops with the OpenTopography team. Iterative development was employed to address these issues and ensure that the integration met all requirements [1].

B. Iterative Development

Frequent changes and feedback necessitated multiple reintegrations. Efficient version control and collaboration were key to managing these iterations and ensuring successful integration.

V. Conclusion

The enhancements to the DesignSafe Recon Portal have significantly improved its utility for researchers studying natural hazards. By integrating OpenTopography data and transitioning to a modern web framework, the portal now offers more detailed insights, better user experience, and greater potential for future enhancements. The project has laid a strong foundation for further development and integration of additional datasets and features.

Acknowledgments

Special thanks to Nathan Franklin, Frank Netscher, Tracy Brown, the entire WMA team, and the OpenTopography team for their guidance and support.

References

[1] OpenTopography, "Opentopography," https://opentopography.org, 2024.
[2] Leaflet, "Leaflet - an open-source javascript library for interactive maps," https://leafletjs.com, 2024.
[3] React, "React - a javascript library for building user interfaces," https://reactjs.org, 2024.
[4] TypeScript, "Typescript - javascript that scales," https://www.typescriptlang.org, 2024.
[5] Tapis Project, "Tapis: Next generation software for distributed research," https://tapis-project.org, 2024.

Machine Learning Edge for Tapis

Sowbaranika Balasubramaniam
Texas Advanced Computing Center
The Ohio State University
Austin, TX, USA
sowbaranika1302@gmail.com

Joe Stubbs
Texas Advanced Computing Center
University of Texas at Austin
Austin, TX, USA
jstubbs@tacc.utexas.edu

Richard Cardone
Texas Advanced Computing Center
University of Texas at Austin
Austin, TX, USA
rcardone@tacc.utexas.edu

Samuel Khuvis
Ohio Supercomputer Center
The Ohio State University
Columbus, OH, USA
skhuvis@osc.edu

Nathan Freeman
Texas Advanced Computing Center
University of Texas at Austin
Austin, TX, USA
nfreeman@tacc.utexas.edu

Tanya Berger-Wolf
Computer Science and Engineering
The Ohio State University
Columbus, OH, USA
berger-wolf.1@osu.edu

Abstract— Edge computing enables devices to deploy Machine Learning (ML) models at the "edge" and process data in near real-time, significantly enhancing response times and detection of sensitive information. However, deploying these models presents several challenges such as the demand for substantial processing power, extensive battery life, and considerable storage and memory capacity. To address these multifaceted challenges, ranging from hardware constraints to data management and complexities of growing ML models, we introduce Machine Learning Edge (MLEdge), which includes the Tapis framework and a robust simulation environment. This environment enables comprehensive testing and optimization of models and deployment strategies, effectively mitigating real-world issues and improving the efficiency and reliability of edge deployments, particularly in camera traps.

MLEdge is a dynamic platform integrated with the Tapis framework, designed to provide a seamless interface for running ML models on edge devices. It facilitates the comparative analysis of various models, datasets, and hardware, optimizing their deployment and performance in resource-constrained environments. MLEdge is developed in collaboration with the Intelligent Cyberinfrastructure with Computational Learning in the Environment (ICICLE) [1] and the Texas Advanced Computing Center (TACC).

Key functionalities of MLEdge include,

1) **User Dashboard**: Users interact with ML Edge Dashboard on Tapis [2] to upload datasets and models, select hardware, configure advanced parameters, and initiate analyses. This process creates a job in Tapis Jobs with specified inputs, which then instantiates the systems controller to communicate with the event engine.
2) **Simulation Environment**: The simulation environment processes images in a manner similar to an edge device. It includes an event engine that employs a publish-subscribe pattern, primarily built with Rust and Python. Plugins are modular components of the application that can generate and consume events. The engine uses ZeroMQ, an open-source messaging library, to manage the delivery of event messages between publishers and subscribers. Input images are processed by the image-generating plugin, which sends the images in binary format to the image-receiving plugin. The images are then passed to the image scoring plugin, where a machine learning model performs detections. Based on a user-specified threshold, the image is either stored or deleted by the Image Store/Delete plugin. All logs are monitored through the Oracle plugin. Real-time monitoring data is transmitted from the edge to the cloud using Cyberinfrastructure Knowledge Network (CKN) Daemon [3] and displayed on an analytics dashboard. Additionally, the power and storage consumption of each plugin are measured.
3) **Reports**: Once execution is complete, the Reports UI provides detailed visualizations of resource utilization and performance metrics, such as accuracy, power consumed, I/O, memory, CPU/GPU usage, and throughput. This allows users to compare the performance of models in different hardware.

MLEdge offers a robust solution to the challenges of deploying ML models on edge devices. MLEdge provides a comprehensive platform for testing, optimizing, and deploying models, enhancing the efficiency and reliability of camera trap deployments in wildlife monitoring and conservation efforts.

Keywords: Edge computing, Machine Learning, Smart Camera Traps, TAPIS, Remote sensors, Simulation Environment, Wildlife monitoring

ACKNOWLEDGMENT

This work was supported by the National Science Foundation's - AI Institute for Intelligent Cyberinfrastructure with Computational Learning in the Environment(ICICLE), Project Tapis: Next Generation Software for Distributed Research and

SGX3 A Center of Excellence to Extend Access, Expand the Community, and Exemplify Good Practices for CI Through Science Gateways under grant agreements OAC-2112606, OAC-1931439 and OAC-2231406.

REFERENCES

[1] D. K. Panda, V. Chaudhary, E. Fosler-Lussier, R. Machiraju, A. Majumdar, B. Plale, R. Ramnath, P. Sadayappan, N. Savardekar, and K. Tomko, "Creating intelligent cyberinfrastructure for democratizing AI," AI Mag., vol. 45, no. 1, pp. 22–28, Mar. 2024, doi: 10.1002/aaai.12166.

[2] J. Stubbs, R. Cardone, M. Packard, A. Jamthe, S. Padhy, S. Terry, J. Looney, J. Meiring, S. Black, M. Dahan, and S. Cleveland, "Tapis: An API Platform for Reproducible, Distributed Computational Research," in Advances in Information and Communication, vol. 1363, K. Arai, Ed., in Advances in Intelligent Systems and Computing, vol. 1363. , Cham: Springer International Publishing, 2021, pp. 878–900. doi: 10.1007/978/-3-030-73100-7_61.

[3] S. Withana and B. Plale, "CKN: An Edge AI Distributed Framework," in 2023 IEEE 19th International Conference on e-Science (e-Science), Limassol, Cyprus: IEEE, Oct. 2023, pp. 1–10. doi: 10.1109/e-Science58273.2023.10254827.

Tapis User Interface for Machine learning Edge

Images are transferred to the datacenter

Edge

- Computing power
- Memory
- Battery
- Storage

Data Center

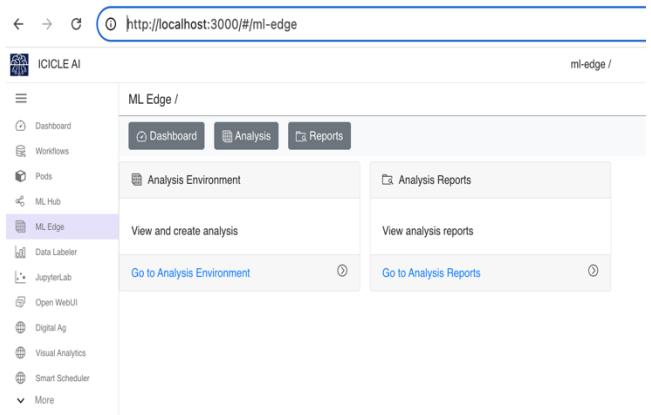
- Network bandwidth to send data to center
- Delay of data analysis

ICICLE DEMOCRATIZING AI

TACC

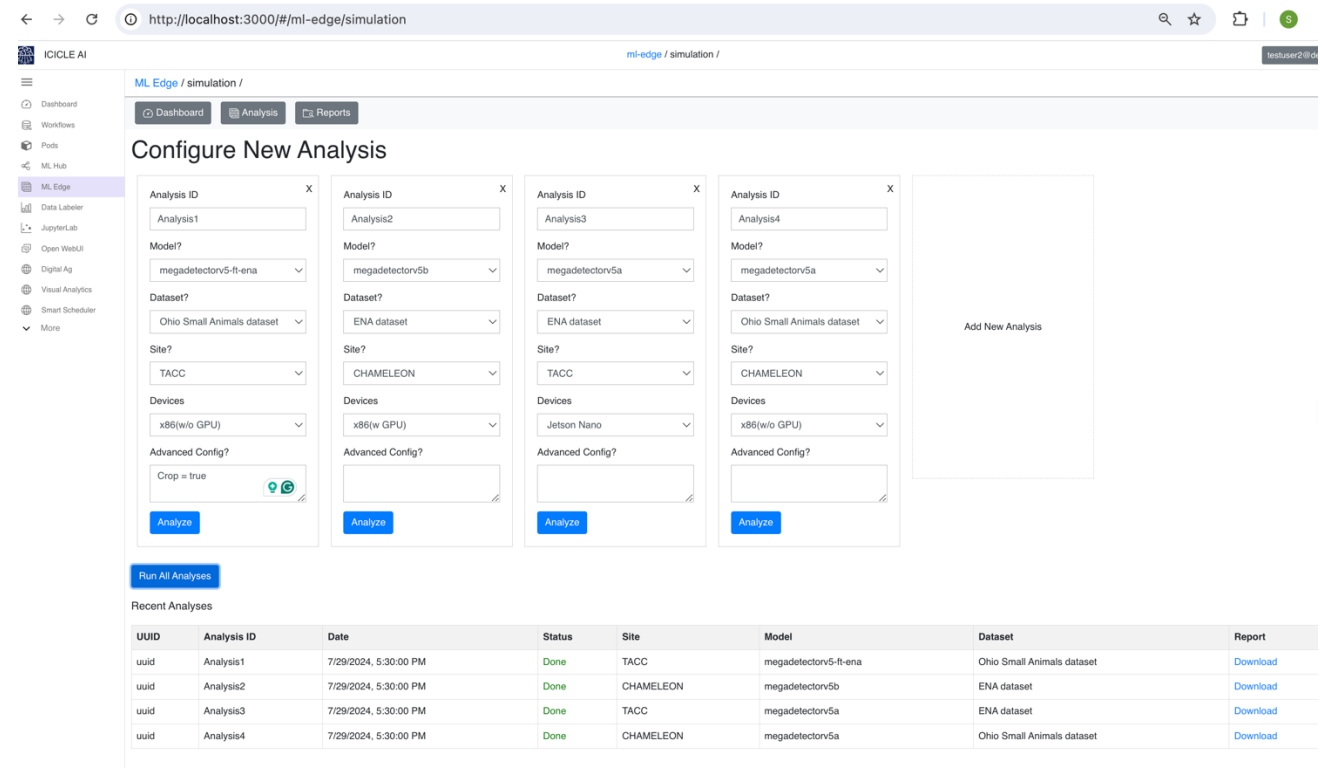
Tapis User Interface for Machine learning Edge

User



Users can inititalize a new analysis by,

1. Selection existing ML models (like MegaDetector)
2. Choosing a dataset available/upload their own
3. Choose the hardware in which they want to execute
4. Add advanced configuration, if any



→ Creates Job request

↓

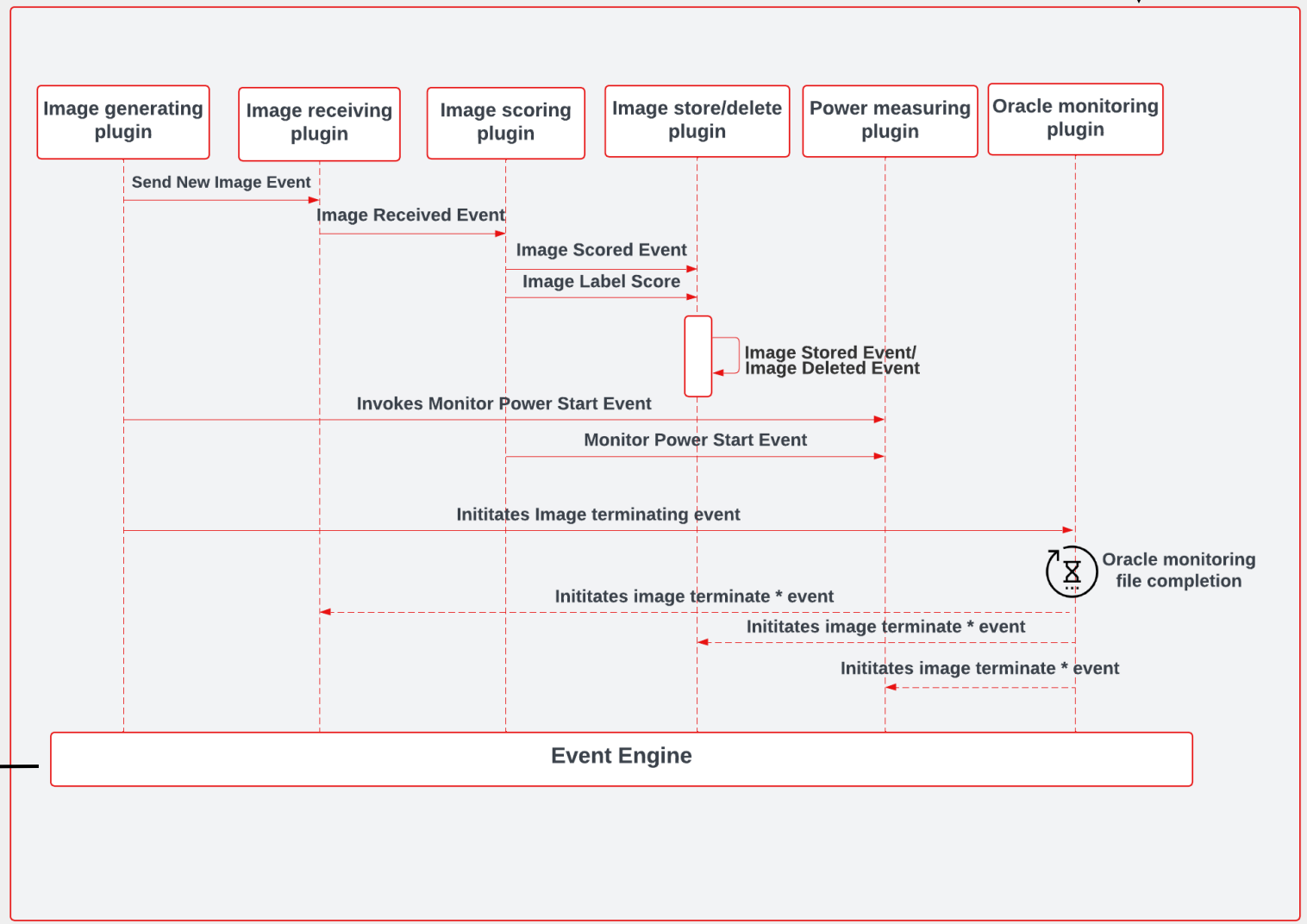
Tapis User Interface for Machine learning Edge

Deploy ML models
on the Edge device

Model Optimization

Resource utilization

- Memory
- Storage
- Power consumption
- Bandwidth



Tapis Streams - Supporting Real-Time Climate Data Monitoring

***Abstract*—Tapis Streams is a production level service, providing REST support for storing, processing, and analyzing real-time streaming data. This poster introduces the newest features of the Tapis Streams service. The latest version of Tapis 1.3 Streams API adopts the latest version of InfluxDB, InfluxDB 2.x, which has built-in security features and supports next generation data analytics and data-event driven processing. This poster also introduces new data Channel Alert actions and archive support. Lastly, we present a Streams service reference user interface as part of Tapis UI, a lightweight browser only, serverless client application that allows interactive access to Stream data supporting a climate monitoring project in Hawaii.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

The proliferation of inexpensive devices in the Internet of Things (IoT) as well as the number of instruments and sensors to observe and measure everything has led to a demands for systems that support storing, processing, and analyzing time-series data. Many scientific use cases based on monitoring require ongoing data processing or special processing/modeling and notification of anomalous or special events that can be identified by processing and computing the data as it arrives. Systems that have been designed for sensors are often aimed at industry, and are either too complex to deploy and maintain or, if hosted, are expensive. This leaves a gap for hosted academic streaming data solutions capable of supporting data event-driven computational workflows. To help address this gap in research cyberinfrastructure, the Texas Advanced Computing Center (TACC) and the University of Hawaii (UH) have developed an open-source, unified middleware API infrastructure platform, Tapis [1], with collaborative features to fill gaps in the existing streaming time-series data landscape, the Streams services [2]. Tapis and the Streams service is currently leveraged in the a Climate Data Portal for the state of Hawaii and the Hawaii Mesonet cyberinfrastructure to support climate science and monitoring..

In this paper, we present the latest architecture of the Tapis Streams services along with new features including: data Channel evaluation methods, Alert actions and reference user interface.

II. Background

This section describes the key concepts involved in the implementation of Tapis Streams API and the API itself.

This work is supported by the National Science Foundation

A. InfluxDB

InfluxDB is a time series database designed to handle high write and query loads. InfluxDB is used in many applications involving large amounts of timestamped data, including DevOps monitoring, application metrics, IoT sensor data, and real-time analytics. The Tapis Streams project has leveraged the InfluxDB and its ecosystem of services, like Kapaictor a real-time stream processing engine, as the time series infrastructure for processing and storing time-series data. The InfluxDB 1.x version is what was leveraged to implement the original Tapis 1.0 Streams APIs. The latest release of the Tapis Streams API has migrated to InfluxDB 2.x which has resulted in some significant implementation changes for the better. Three major changes include moving the once separate real-time processing service into InfluxDB, adopting Flux as the supported scripting language for queries and tasks, and making the Influx database into bucket or name-spaced structures. These changes are discussed in more detail in the Implementation Section.

B. Tapis v3 Streams

Tapis version 3 (v3) provides an enriched set of open-source, hosted Application Program Interface (API) platform for distributed computation that enables researchers to manage data and execute codes on a wide range of remote systems, from high-speed storage and high-performance computing systems to commodity servers.

The Tapis Streams API has been developed to support real-time streaming data workflows with storage, retrieval, and analysis of temporal sensor data. The Streams API is built using the Python Flask web framework and interacts with other Tapis Services such as Actors, Jobs, Security Kernel, Meta, Tenants, and Tokens services as shown in Figure 1. Streams resources are hierarchical and based on the Cloud Hosted Real-time Data Services(CHORDS) [3] resource model, for example, Project is at the top level in the hierarchy which contains important information such as project description, principal investigator, owner, metadata about the project. Next in the hierarchy is a Site. A site is a geographical location with spatial coordinates such as latitude, longitude, and elevation, where the physical hardware for remote sensing is located. A project can have multiple sites and the geo-spatial coordinates associated with the sites can be used to search data related to that site. The physical hardware where multiple sensing devices are embedded is next in the hierarchy and is known as an instrument. Each site can host multiple instruments and they can be identified with their unique ids. Individual

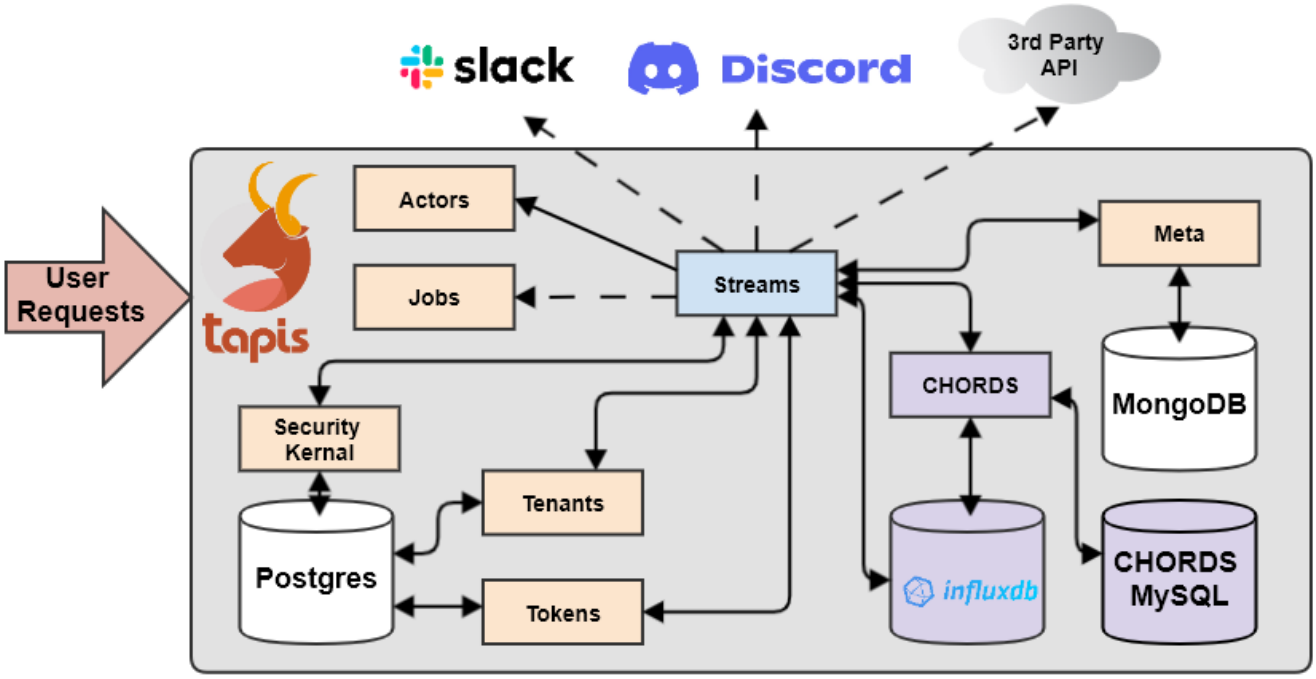


Fig. 1. Tapis Streams API Architecture and New Channels Action Integrations (Slack, Discord, 3rd Party Web-hooks, Jobs) indicated with dotted lines. Orange boxes are other Tapis API services leveraged by Streams. Streams specific infrastructure dependencies are shown in purple.

sensors are referred to as variables, which sense physical parameters such as temperature, humidity, rainfall, etc., and these measurements are stored in the InfluxDB time-series database. All the resources can be accessed only by authorized users and individual user roles.

The Streams API is deployed on an on-premise Kubernetes cluster (Fig 1.) hosted at TACC along with other production-grade Tapis services. The Streams API deployment consists of four primary components: the Python API, the CHORDS server, and two databases: InfluxDB for time-series measurements and MySQL which CHORDS uses. CHORDS and InfluxDB services are also deployed as individual services in the same cluster. Kubernetes ConfigMaps and secrets objects are used to configure the deployments. A Tapis deployer tool, developed at TACC is used to automate the creation of ConfigMaps and secrets, persistent volume claims, and to start the entire Streams API stack. Every user request to access Streams resources first goes through the Tapis Security Kernel for authorization and authentication check to ensure that the user has the necessary role to perform the requested action on the specified Streams resource. Tokens service provides a signed service JWT, which lets the Security Kernel and Metadata service know that request is coming from an authentic source, i.e., Streams service. Metadata service provides a backend MongoDB for the Streams API, which stores all the metadata associated with the Streams resources.

C. Tapis UI

To facilitate the usage of the Tapis APIs, Python and TypeScript packages were developed. These packages provide wrapper functions for submitting requests to Tapis. Additionally, an online portal for interacting with Tapis via a user interface, Tapis UI, was developed. This interface was created using React, an open-source JavaScript framework for creating web applications, and leverages the Tapis TypeScript library for dispatching user actions to the Tapis APIs. A component for interacting with the Tapis Streams API was included in this portal. More details on Tapis UI are discussed in the later sections.

III. Tapis UI for the Streams API

The Tapis UI Streams interface allows users to view data stored by the Streams API. Projects, Sites, and Instruments are listed in a hierarchical interface. Users are initially presented with a list of projects. Clicking on a project will list the sites associated with that project, and clicking on a site will list the instruments associated with that site. Once a site is selected the measurements for each variable tracked by that instrument are displayed.

The displayed measurement data is grouped by variable name. Each group provides a listing of the measurement values and a timestamp for when the measurement was taken. Additionally, a graph representing a time series of the measured values is provided (Figure 2). Large numbers of values are collapsed by default, displaying only the first and last two

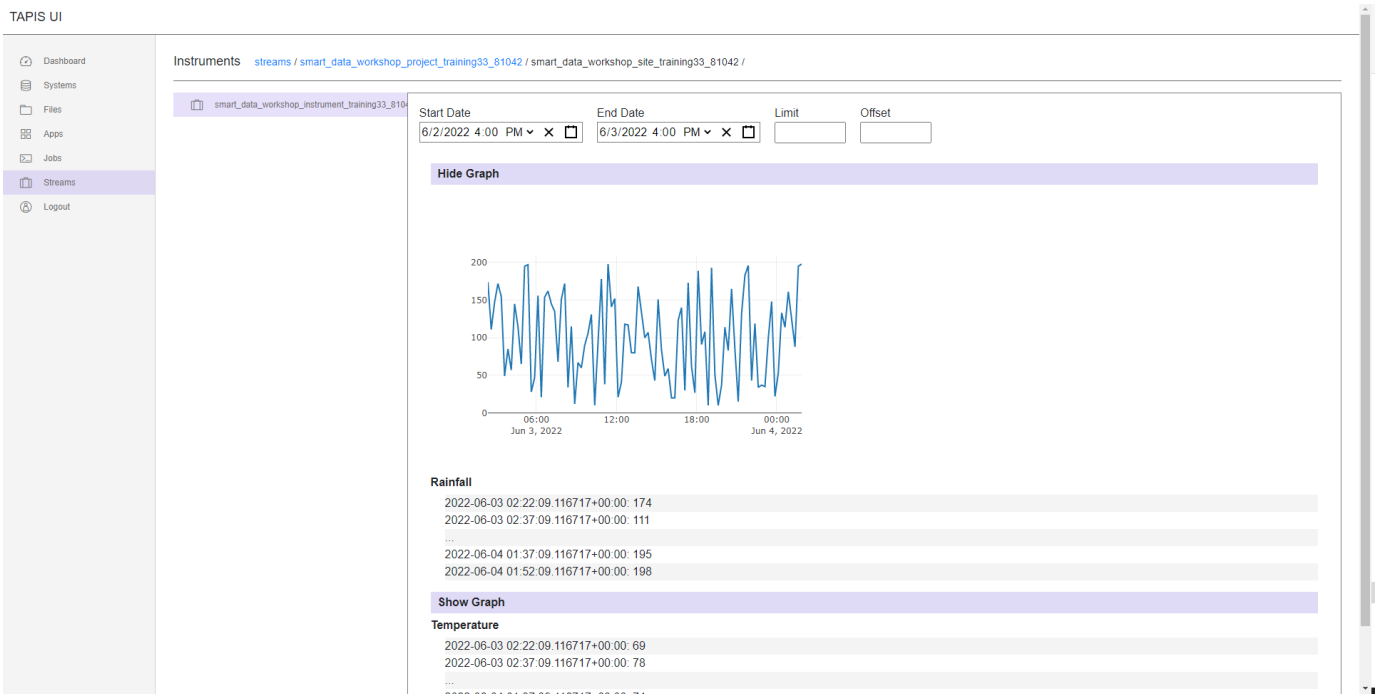


Fig. 2. Tapis UI view for the Streams API. Displaying a set of measurements for rainfall and temperature over a 24-hour period.

measurements. This can be expanded by clicking on the block of values.

The set of returned measurements can also be limited using a set of filters displayed at the top of the measurements panel. The available parameters are start date, end date, limit, and offset. Setting a start or end date will limit the returned measurements to the specified range based on their timestamp. The limit field specifies a maximum number of values to be returned. The offset field specifies the first value to be returned. For example, an offset of 5 will skip the first four measurements and return values starting at the fifth measurement.

This portal provides a simple pre-developed dashboard for users to view data being pushed into the Streams API. This can limit the need for additional developer overhead for generating basic visuals or monitor and validate data streams.

IV. Conclusion

In conclusion this paper has presented the updates to the Tapis Streams design and implementation that enhanced security, utility and access while maintaining the integrity of the specifications. These new enhancements allow the Tapis Streams API to better serve researchers through enhanced notifications, integrations and basic data access and visualization of Stream's data. The Tapis Streams API will continue to evolve with additional actions to include advanced search capabiites, share-able pre-authenticated data links and ontology support for rich metadata.

V. Software Availability

The source code for the Tapis Streams API is available on GitHub at https://github.com/tapis-project/streams-api. Source code for the Tapis-UI with streams can be found https://github.com/tapis-project/tapis-ui

Acknowledgements

This work is supported by the National Science Foundation Office of Advanced CyberInfrastructure - Tapis Framework [#1931439, #1931575] and the RII Track-1: Change Hawaii: Harnessing the Data Revolution for Island Resilience NSF OIA #2149133.

References

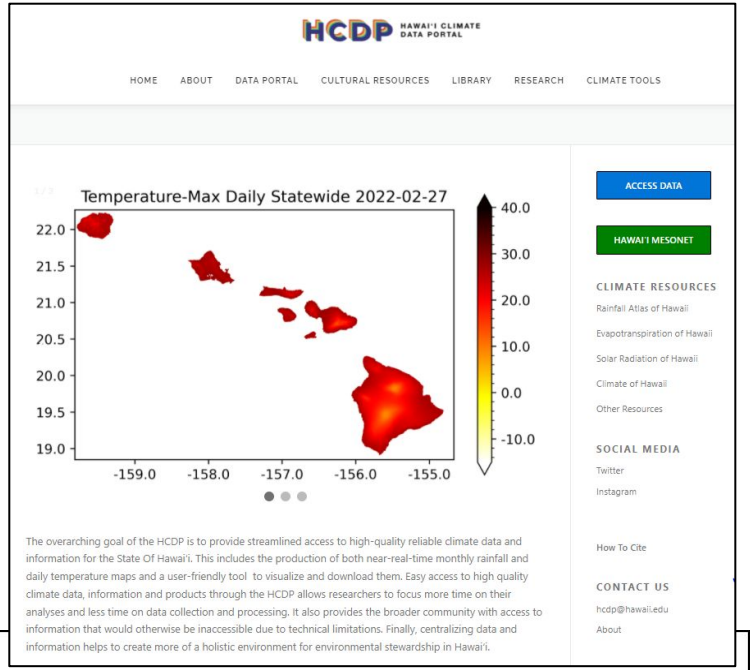
[1] J. Stubbs *et al.*, "Tapis: An api platform for reproducible, distributed computational research," *Future Generation Computer Systems*, 2021, accepted.

[2] S. B. Cleveland, A. Jamthe, S. Padhy, J. Stubbs, S. Terry, J. Looney, R. Cardone, M. Packard, M. Dahan, and G. A. Jacobs, "Tapis v3 streams api: Time-series and data-driven event support in science gateway infrastructure," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 19, p. e6103, 2021.

[3] B. Kerkez *et al.*, "Cloud hosted real-time data services for the geosciences (chords)." Geoscience Data Journal, 2016, pp. 2–4.

What is the Hawai‘i Climate Data Portal (HCDP)

- **A Place** to get climate data and information
- **A Tool** that a allows us to explore the past, monitor the present, and project the future
- **A Portal** to other places and data sources
- **An Opportunity** to learn, to education, to network, and to share



https://hawaii.edu/hcdp

HCDP HAWAI‘I CLIMATE DATA PORTAL

HOME ABOUT DATA PORTAL CULTURAL RESOURCES LIBRARY RESEARCH CLIMATE TOOLS

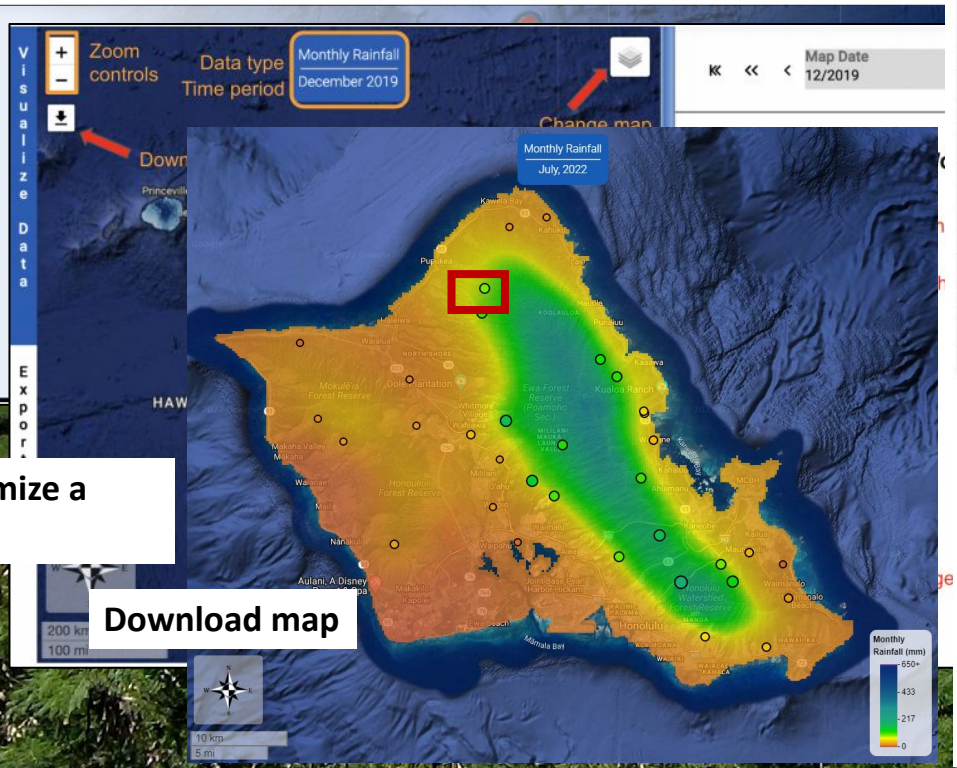
NSF #OIA-2149133

HCDP HAWAIʻI CLIMATE DATA PORTAL

HOME ABOUT DATA PORTAL CULTURAL RESOURCES LIBRARY RESEARCH CLIMATE TOOLS

Visualize Data

Visualize available climate data in the portal.



Customize a Map

Download map

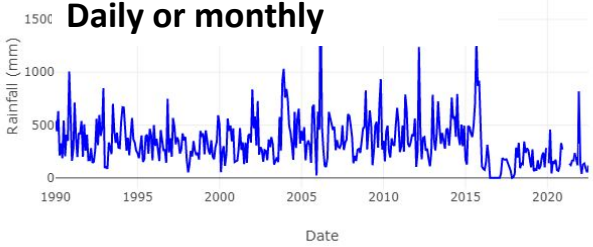
Name	Station ID	Island
PunaluuRainGageatalt	884.4	O'ahu
SCHOFIELD BARRACKS	858	O'ahu
Poamoho Exp Farm	855.3	O'ahu
DILLIINGHAM	843.7	O'ahu
MakahaRainGage	842.1	O'ahu
Waihee Pump	839.8	O'ahu
WAIAHOLE STREAM NEAR KAHALUU 2N	837.12	O'ahu
Palisades	835.2	O'ahu
Waiawa CF	834.13	O'ahu
SCHOFIELD EAST	828	O'ahu
Waipio	824.2	O'ahu
Mililani	820.6	O'ahu
WHEELER ARMY AIR FIELD 810.1	810.1	
WAIANAE VALLEY	803.2	
Waimanalo Nonokio	795.3	
BELLOWS AFS AT WAIMANALO	793.2	

Station Metadata	
SKN (Station ID)	884.4
Name	PunaluuRainGageatalt
Observer	USGS
Network	USGS
Island	O'ahu
Elevation (m)	73.15
Latitude	21.56
Longitude	-157.9
NWS ID	PNSH1
NESDIS ID	DD989652
Value	120.39

Get Station Meta Data

Adjustable time series plots

Daily or monthly



Gateway Poster

Text Mining and Sentiment Analysis of YouTube Videos: New Analytics Features for the "Social Media Macroscope" Science Gateway

Abstract:

This poster proposes a computational approach that supports social media analytics that can be included in the science gateway, Social Media Macroscope (Wang et al., 2023; Yun et al., 2019). This research applies text mining and sentiment analysis on historical and current opinions from YouTube videos regarding cryptocurrencies to construct a bidirectional Recurrent Neural Network model. The model predicts whether sentiments are positive or negative, aiming to assist and support investment decisions in cryptocurrencies. This analytics feature can be incorporated into the science gateway, Social Media Macroscope. Why is this feature important? Currently, cryptocurrencies are regarded as highly potent and popular assets with volatile market prices driven by market mechanisms, lacking central authority or intermediary. They facilitate direct transfer of value or data across the internet, recording all transactions in an immutable manner, ensuring flexibility, high security, and increasing accessibility and interest. However, the vast variety and high market volatility of cryptocurrencies are influenced by market demand, real-world applications, regulatory frameworks, competition, and news. This high investment risk complicates decision-making for investors on when and which cryptocurrencies to invest in. Overall, this poster contributes to the knowledge on science gateways by expanding research and capability of science gateways for the social sciences in general, and social media analytics specifically.

Keywords: text mining, sentiment analysis, social media macroscope, cryptocurrency, social media analytics

References

Wang, C., Kim, Y. W., Kooper, R., & Yun, J. (2023, October 30). SMILE: A User-Friendly Science Gateway for Social Media Research and Collaboration. Science Gateways 2023 (SG23), Pittsburgh, PA. https://doi.org/10.5281/zenodo.10028454

Yun, J. T., Vance, N., Wang, C., Marini, L., Troy, J., Donelson, C., Chin, C. L., Henderson, M. D. (2019). The Social Media Macroscope: A science gateway for research using social media data. Future Generation Computer Systems. doi:10.1016/j.future.2019.10.029