Better Data Discoverability in Science Gateways

Supun Nakandala, Sudhakar Pamidighantam, Suresh Marru, Marlon Pierce
Science Gateways Research Center, Indiana University
{snakanda, pamidigs, smarru, marpierc}@.iu.edu

ABSTRACT

Science gateways primarily focused on remote job execution management generate domain specific output data mainly readable by application specific parsers and post processing utilities. For example, computational chemistry data outputs encode molecule information, convergence of the simulation and energy values. Such domain-specific information is non-trivial to search in a generic fashion. It is thus desirable to add a wide range of application-specific and user-specific post-processing features that may include remote executions of scripts and smaller applications that don't require scheduling on clusters. It is also desirable to support integrations with searching, indexing, and general purpose data analysis and mining tools provided by the Apache "big data" software stack. As gateways become tenants to general purpose platform services, providing a general purpose infrastructure that enables these application specific post-processing steps is an interesting architectural challenge. Furthermore, it is desirable to share results from the post-processing and indexing. In this paper, we discuss how we have incorporated a new automated application output indexing system for the SEAGrid Science Gateway using Apache Airavata that will parse and index generated output for easy querying. We also examine data sharing and automated data publication so that another user can reuse the results without running an already executed experiment and hence reduce resource utilization.

1. INTRODUCTION

SEAGrid Science Gateway [4] and its predecessor, GridChem Science Gateway, have been serving computational chemistry communities and related domains for more than 10 years. Together, SEAGrid and GridChem have generated more than 8TB of application specific archived outputs. As the execution exceeds hundreds of experiments by a given users, managing and browsing through them becomes non-trivial. The search capabilities are limited by user provided limited metadata, but if additional metadata can be extracted from application outputs, users can find value in their previous runs and also share them with theirs in a meaningful way. Efforts such as [2],[1] are highlighting the importance of such extracted metadata. Without burdening the users, we have added new capabilities ensuring:

- Metadata extraction be done automatically (so that a user is not required to manually feed them to the system).
- Users search capabilities to include both content of the data in addition to user provided experiment name and descriptions.

Considering these features, we have developed an automated metadata extraction system that can extract metadata from the data by running sets of configurable parsers. In our system, we have focused on parsing output data generated from several computational chemistry applications including Gaussian, Molpro, NWChem, and Gamess. However, the concepts and design of this system can be widely applied to any type of application and data. In the following sections we will describe the high level design of the new system and details on the implementation.

2. DESIGN & IMPLEMENTATION

The SEAGrid Science Gateway uses the Apache Airavata science gateway framework [3] for its remote job execution and management tasks and WSO2 Identity Server for user identity management. Airavata builds on the component based microservices architecture, and hence it is easy to add new functional components that relies and builds on top of the existing components. In Figure 1 we have shown how the new parsers and search functionality is seamlessly integrated within existing Apache Airavata framework.

Within SEAGrid, the user provided input data and the corresponding generated output data archived within the SEAGrid Data Store. When a user runs an experiment, the input files are staged into a folder in this server, which will be then copied to the remote computational resource by Airavata. After the remote application completes the generated output, data are again moved back to this data store by Airavata. Status information about different states of experiment execution can be obtained by subscribing to the AMQP based Airavata message bus. The message bus is the point of extension that we use to enable post-processing pipelines.

We have deployed a message listener daemon, the Airavata Message Listener in Figure 1, which is subscribed to all experiment status change messages in the SEAGrid Gateway. Once an experiment completion message is received it will again query Airavata's registry component to get more details about the completed experiment such as application type and output data path, and generate a data parsing work object. The work object is then added to a work queue. At the other end of the queue, worker nodes are deployed on multiple machines that have local access to the gateway data store through NFS mounting. Worker nodes parse the output data and extract metadata, as described below. The extracted metadata is then formatted into a JSON format and indexed into a MongoDB database. We have also developed a web-based data portal which enables users to query on data based on the extracted attributes. For example in SEAGrid case users can now search based on molecular structures, energy values, and several other extracted domain specific attributes. In the default configuration only

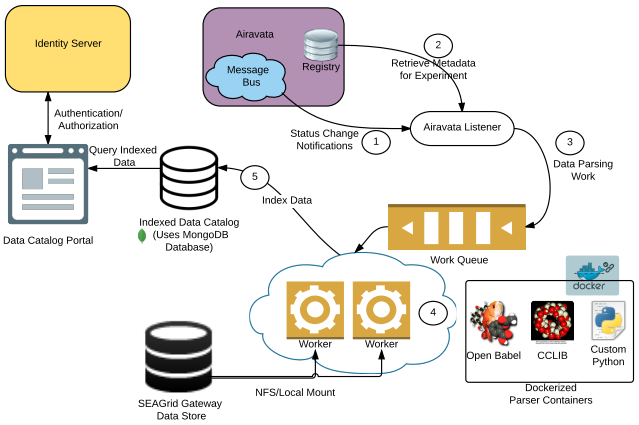

Figure 1: SEAGrid data cataloging system

the owner of the data has access to the extracted metadata, but the owner can make the data publicly shared so that others can have access.

A particular parsing and extraction pipeline is provided to workers by configuring a set of parsers. For each application, multiple parsers can be configured. Extracted metadata from each of the parser is then merged into one JSON document and indexed in MongoDB. Parsers are containerized into docker containers and can contain any type of parsing logic written in any language. For example, in SEAGrid, we mainly use python-based parsers that also rely on system-installed libraries such as Open Babel and CCLib (Computational Chemistry Library). The advantage of containerizing the parsing logic is that the containers can be easily executed on any machine without installing complicated and incompatible libraries on the host server. Another advantage of dockerizing the parser logic is that whenever changes need to be done for the existing parsers, those changes can be centrally published using DockerHub, and all workers will thereafter use the updated docker images.

During our initial development, we noticed that it is hard to provide a fixed schema for the metadata model, as different parsers may extract different metadata values. Coming up of with a fixed set of metadata attributes prevents the extensibility of existing parsers and addition of new parsers. Therefore we format the extracted metadata from parsers using a key-value format into one aggregated JSON document. This schema-less approach enables easy extensibility of the data cataloging system.

It is possible for the parser logic to change in existing parsers or for new parsers to be added; this can produce inconsistencies in the metadata based on the version of the post-processing pipeline. Our current solution is to rerun the parsers on the already catalogued data. This can be done by replacing the Airavata Message Listener component combined with an Airavata Registry scanner or a file system scanner that will generates data parsing work units and add them to the worker queue. This listener creates "new" events on the Airavata message bus from the archived data.

3. FUTURE WORK

The data parsing (or post processing) tasks for SEAGrid described here are done in a prototype environment external to Airavata's main job management features. We are integrating these extensions into the Airavata core so that it can also be used to execute these data parsing jobs. In this case jobs will be self containerized docker images similar to the current implementation. The advantage of using Airavata instead of custom built data parsing infrastructure is that we can leverage the scalability and fault tolerance aspects that are already available in Airavata. We also will increase the number of parsers available in the system so that outputs from many applications can be parsed. Our goal is to make the features described here into a first class capability within Airavata and expose these features to additional gateways. Finally, as we move these features into production for SEAGrid, we will merge the currently separate data portal with the SEAGrid main portal as a top level gateway functionality.

Another aspect that we are examining is the integration of a data publishing platforms to SEAGrid Science Gateway. There is an increasing trend among academic publishers to require data to be published (accessible) in addition to methodologies and results used in the scientific process. There are several such existing platforms that allows users to upload data and publish them; after publishing, a persistent document object identifier (DOI) is returned which can be used to cite/locate the data. What we are trying to achieve in SEAGrid is to make the data publishing step an integral part of the gateway functionality such that it seamlessly integrates with other gateway operations. In order to achieve this, we are developing connectors to existing data publishing platforms from the SEAGrid gateway portal. Finally, as we move these features into production for SEAGrid, we will merge the current proof of concept data portal with the SEAGrid portal as a top level gateway functionality.

4. ACKNOWLEDGEMENTS

This work was partially supported by NSF ACI award #1339774, "Collaborative Research: SI2-SSI: Open Gateway Computing Environments Science Gateways Platform as a Service (OGCE SciGaP)".

References

[1] Sarah Allanson Cashman et al. Mining available data from the united states environmental protection agency to support rapid life cycle inventory modeling of chemical manufacturing. *Environmental Science & Technology*, 2016.

[2] Matthew J. Harvey et al. Digital data repositories in chemistry and their integration with journals and electronic notebooks. *Journal of Chemical Information and Modeling*, 54(10):2627–2635, 2014.

[3] Suresh Marru et al. Apache airavata as a laboratory: architecture and case study for component-based gateway middleware. In *Proceedings of the 1st Workshop on The Science of Cyberinfrastructure: Research, Experience, Applications and Models*, pages 19–26. ACM, 2015.

[4] Sudhakar Pamidighantam et al. Community science exemplars in seagrid science gateway: Apache airavata based implementation of advanced infrastructure. *Procedia Computer Science*, 80:1927–1939, 2016.