MPO: A System to Document and Analyze Distributed Heterogeneous Workflows

Conference Paper · June 2016
DOI: 10.1007/978-3-319-40593-3_14

10 authors, including:

Kesheng Wu
Lawrence Berkeley National Laboratory
255 PUBLICATIONS **4,416** CITATIONS
SEE PROFILE

Arie Shoshani
Lawrence Berkeley National Laboratory
211 PUBLICATIONS **5,955** CITATIONS
SEE PROFILE

John Wright
Massachusetts Institute of Technology
187 PUBLICATIONS **1,208** CITATIONS
SEE PROFILE

This an extended version of https://doi.org/10.1007/978-3-319-40593-3_14

MPO: a System to Document and Analyze Distributed Heterogeneous Workflows

K. Wu[3], E. N. Coviello[1], S.M. Flanagan[1], M. Greenwald[2], X. Lee[1], A. Romosan[3], D.P. Schissel[1], A. Shoshani[3], J. Stillerman[2], J. Wright[2]

[1]General Atomics, P.O. Box 85608, San Diego, CA 92186-5608, USA
[2]Massachusetts Institute of Technology, Cambridge, MA 02139, USA
[3]Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA

Abstract. Large scientific experiments and simulations produce vast quantities of data. Though smaller in volume, the corresponding metadata describing the production, pedigree, and ontology, is just as important as the raw data to the scientific discovery process. Metadata could be automatically captured by workflow management systems or specially designed operating system plug-ins. However, there are many cases where these automated systems are insufficient; in which cases the critical metadata must entered manually into the computer systems or it will be lost. To automate the metadata capturing in these cases, we develop a generic metadata capturing and analysis system called MPO (Metadata, Provenance, Ontology). It could seamlessly integrate with most data analysis environments and requires a minimal amount of changes to users' existing analysis programs. Users have the full control of how to instrument their programs to capture as much or as little information as they desire. Users can further design, control and evolve the ontology used for describing the workflows. Once captured in a database system, the workflows can be visualized and studied through a set of web-based tools. In large scientific collaborations where the workflows have been built up over decades, this ability to instrument a complex existing workflow and visualize the key interactions among the software components is found to be tremendously useful by users. In this paper, we describe the MPO concepts, its software architecture as well as the recent additions to enable ontology evolution and querying of workflows. We also report on our deployment experience on a couple of applications.

1. Introduction

Datasets collected from scientific experiments and generated from computations typically go through numerous analysis steps on the path toward developing scientific knowledge. These processes of data generation, conversion, manipulation and transformation are often formalized and codified into sequences of steps known as workflows; an example from a fusion simulation is given in **Error! Reference source not found.**. In this process, we distinguish the raw data from the metadata about the raw data. Though the metadata is typically much smaller in volume than the raw data, it contains critical information such as how the raw data is organized, where the data is from, and how the numbers and strings data are to be interpreted. In this work, we pay particular attention to two specific type of metadata known as provenance and ontology. Provenance is the metadata that describes how a dataset is derived or processed. It is of particular interest here because it is important for scientists to reproduce the data analysis and to study the data analysis process [1]. Ontology is a formal naming and definition of the types, properties, and interrelationships of the entities for a particular domain of science [2, 3].

When all steps of a data analysis process is performed within a single workflow management system, the workflow management system often has a way of capturing the provenance information [4, 5]. However, there are many active research collaborations with decades of history and large collections of workflows that are not on any of the modern workflow management systems. In addition, a large workflow might involve an extended collaboration and span a number of different computer systems, where no single workflow

management system could reach all of the disparate parts. In such a case, scientists have to manually enter the critical pieces of metadata including the provenance information. Manually entering metadata requires scientists to break their attention on the data analysis process and potentially decreases their productivity, which diminishes the chance that metadata will be entered in a timely manner if at all. Furthermore, there is no easy way to enforce a consistent ontology in a distributed environment. Inconsistencies in terminology used in describing the workflows and data products could cause confusion in their later uses, and reduce the value of the data products, which further reduce the motivation for users to enter metadata about their work. Clearly, automating the metadata capturing and using consistent ontology are essential to address these difficulties. The key challenge is to do these on an arbitrarily complex workflow in a distributed environment.

Figure 1: Example data preparation workflow to execute the GYRO code <https://fusion.gat.com/theory/Gyrooverview>.

Our answer to the distributed metadata capture problem is a system that works with any computing platforms, captures information from workflows executed anywhere, and requires a minimal amount of modifications to the existing workflow components. The system is known as MPO, which is a shorthand for Metadata, Provenance, and Ontology[1]. This paper summarizes the current status of the design, development and testing of the system. Following a study of different models for data tracking, cataloging, and integration across a broad range of scientific domains, we designed a set of tools to instrument scientific workflows, capture the associated metadata, and display the information for interactive study of the workflows [6]. The MPO software consists of a data model, an API for capturing information, a database for storing the captured information, and a web service for analyzing the captured information [7]. Workflows are represented as directed acyclic graphs, providing explicit information about the relationships between workflow data and actions. This graphical representation is accessible anywhere through a web front-end.

The high-level design of MPO and the initial prototype system have been described earlier [6-9]. In this paper, we briefly review the key concepts and then describe the new features that enable ontology evolution and querying of workflows. Furthermore, we will examine a few of the recent use cases from different application domains to demonstrate the generality of the tools.

---

[1] MPO project documentation and software are available at <https://mpo.psfc.mit.edu/>.

[2] SQLAlchemy is available at http://www.sqlalchemy.org/

2. Related Work

The need for automatically capturing metadata alongside the raw data is recognized by both academics and businesses for many years [10-12]. There are a number of systems with the goal of easing the creation of contextual metadata and provenance information, and helping scientists to quickly find answers for questions on any data element related to their research. The MPO system shares this high-level goal with all of them. In this section, we briefly review some key representatives of these related systems and explain the essential differences between these systems and MPO.

When a workflow system is used to manage an analysis process, the metadata about a workflow can be captured by the workflow management system itself. For example, the VisTrails workflow system captures the metadata and provenance so that the workflows could be modified and manipulated easily [13]. However, this approach is restrictive because there are many analysis workflows outside of workflow management systems. Furthermore, there are many workflows that spread across multiple disjoint computer systems, but the workflow management systems would not in general go across different computer systems. In these cases, the metadata captured by the different workflow management systems would also be disjoint and could be hard to use.

A way to connect the disjoint workflow management systems is to develop a globally connected computing infrastructure. A widely used version of such an infrastructure is the Open Science Grid [14, 15], which defines a set of standards for computers to interact with each other. The infrastructure also contains a number of metadata services that are independent of workflow engines [16, 17]. However, such metadata services still rely on the Grid infrastructure and a set of common vocabularies. The MPO system captures metadata based on user-defined ontology, so that users can have more control over what is recorded by the system.

The recent development work on MPO primarily focuses on two topics: ontology evolution [18-21] and provenance search [13, 22, 23]. Next, we briefly review a few representative articles from each of these two topics.

In one of the earliest research papers on the topic of ontology evolution [18], the authors outlined a six-step process: (1) capturing, (2) representation, (3) semantics of change, (4) implementation, (5) propagation, and (6) validation. Many research articles concentrate on a specific aspect of ontology evolution specified here, and there are also many that focus on application use cases. For example, Noy et al [19] address the issue of automatically detecting the implications of ontology changes, while Kondylakis and Plexousakis [20] address the need to automatically rewrite the queries on the ontology terms. Zablith et al [21] provide a survey of the literature on the topic of ontology evolution. In our work, instead of supporting ontology evolution in a comprehensive way, we take a practical approach of supporting a minimal set of functions to support the modification of ontology and propagating the changes.

The ability to search provenance is recognized to be something useful in many applications [13, 22, 23]. The survey article by Freire et al [22] lists querying capability as one of the three components of a provenance management system. This survey mentions a number of special storage systems and search methods for provenance. In addition, it also lists a number of systems that utilize provenance to enable and enhance their operations. For example, VisTrails [13] utilizes provenance information to assist users with modifying workflows and learning about analysis process. In the similar spirit, the MPO system also supports queries on metadata and provenance to assist the users in their study of the work-

flows as part of our support for analysis and visualization of workflows reported to the MPO database. This workflow analysis tool is currently a hosted web application that could be accessed anywhere, thus scientists can study their workflows with nothing more than a web browser.

3. Basic Concepts

The MPO system documents scientific research activities by tracking experimental and computational workflows. We aim to organize metadata from a wide range of applications, and support a diversity of analysis operations; we need to define a basic common vocabulary. We consider these as the basis of an MPO data model. Here are the definitions of the basic terms:

Data Object – A unit of information related to a scientific activity or research. The size of this unit is application dependent. It could be a large dataset, a single value, a graph, or a collection of research paper. In most case, it is useful to refer to it as a single object because it is either produced or consumed by a computer program as a unit. The Data Object can be stored in a file or in a data store. MPO keeps a pointer – in the form of a URI (Uniform Resource Identifier) – that uniquely identifies the data and its access methods. It keeps track of Data Objects in two ways. First is the general information about the Data Objects themselves, such as URI, general comments, and metadata. Second is the specific use of the Data Object in a workflow, such as comments about the usage of the Data Object within a workflow.

Activity – Anything that creates, moves, or transmutes data from one form to another. An Activity could consume multiple input Data Objects, and produce one or more Data Objects as output. Examples of Activities include data importing, pre-processing, input preparation, executing codes, data storage, post-processing, plotting, and data exporting. All of our existing Activities produce and consume Data Objects stored on disks; however, there is nothing preventing our system from working with Data Objects in memory.

Connection – The causal link between multiple Activities and/or Data Objects.

Workflow – A series of connected Data Objects and Activities organized as a Directed Acyclic Graph (DAG). A Workflow shows the individual steps in the processing chain and the parent-child relationship among its elements (Data Objects and Activities). Our current definition of a Workflow could not include loops. This limitation reduces the complexity of the analysis tool to be developed. We plan to explore the options of supporting loops in the future.

Collection – A group of related entities. A Collection may contain any number of Data Objects and Workflows. A Collection may also include other Collections. An element of one Collection could be shared among multiple Collections. Relevant use cases are: 1) A series of simulation runs in a parameter scan; 2) Multiple computational data analysis workflows and associated data used in a published paper.

Metadata – A name-value pair associated to a Data Object, Activity, Workflow, or Collection. Examples of Metadata include data of most recent update to a Data Object and a structured note about an algorithm used in an Activity.

Comment – Free-form text (including hypertext) information associated with a Data Object, Activity, Workflow, or Collection. It is also call annotations in other workflow management systems. In MPO, Comments can be added recursively. Comments are unstructured and may come with a few fixed attributes, such as user ID and timestamp of creation.

Provenance is the lineage of Data Objects. It traces the path of a Data Object from creation through every transformation. Every time a workflow is executed, an instance of that execution is generated. That instance represents the provenance, which includes the Data Objects and parameters used as input for each step, the Data Objects and parameters generated as output, and the sequential relationships between the steps. Extensive information, such as where a piece of data came from, how it was created, why and by which Activity, are captured as the contents of Provenance.

Ontology is a structure for capturing terms used to describe object properties in a domain of research. This is also referred to as controlled-vocabulary or classification structures. This common vocabulary allows domain scientists to express workflows in a consistent way. Ontology is usually represented in tree structures, where the leaves contain "narrower terms", and the higher-level elements as "broader terms." The tree of terminology will be different for different domain of application, and needs to be developed in cooperation with application scientists.

The MPO entities described above are the basis for automatically generating visual representations and describe relations among multiple Data Objects, Activities, and Workflows. They are also essential in creating a software framework for documenting Provenance and Ontology.

4. System Architecture

In this section we review the basic architecture and key components of MPO software. We will describe the key new features in the next section.

4.1. MPO Architecture

We designed the MPO system to support a variety of data, processing programs and complex connections among these data objects and programs. The computational codes can be in a variety of languages (FORTRAN, IDL, C/C++, Python, shell scripts, Matlab), and they may run a variety of computational environments (e.g., operating systems, interconnects, software libraries). The computer hardware can be a laptop, desktop, computer cluster, and even supercomputers that are physically distributed in different corners of the world. Furthermore, the input and output data formats can be very different as well, for example, MDSplus [24], HDF5 [25], JSON [26], and CSV(comma separated values) [27]. These diverse set of data objects, programs and computers could be orchestrated to perform as imaginative dance as human mind could conceive them.

MPO is designed to allow scientists to capture information about their divergent workflows from anywhere. We design the MPO software system as multi-tier web services. It defines a RESTful [28, 29] API that can be easily accessed from a variety of programming languages for instrumenting MPO client calls. The current architecture of the MPO system and its main components are shown in Figure 2.

The building blocks of the MPO system are: 1) Database; 2) API Server and Event Server; 3) Interactive UI server; 4) Clients. The heart of the system is a set of web servers: API Server, Event Server, and the Interactive UI Server. The API server communicates with a Database system to store the persistent data. A client may communicate with the API server or the UI server. Those that directly communicate with API server are "Native Application" clients, while those that communicate with the UI server are web clients.

4.2. Database

The MPO database is responsible for storing all persistent information. Its schema is based on MPO entities discussed in Section 3, and the MPO entities (Data Objects, Activities, Con-

nections, Workflows, Collections, Metadata, Comments, and Ontology) are represented with database tables. There are also several additional tables including the users table. The table representing the DAG structure is a Connectivity table. The content of this table is essential to support the concepts such as Workflows and Provenance.

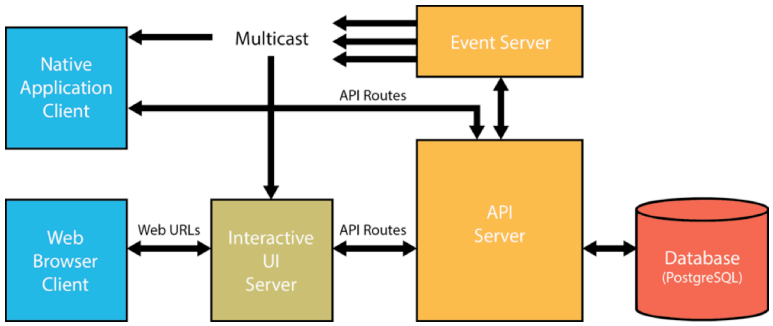

Figure 2: the main components of MPO system.

Currently, PostgreSQL, an open source relational database system, is chosen for the database server. However, the MPO database schema design is general enough that the database server can be replaced with another database system. The generality of the schema is helpful in separating the database implementation from design of user API and other functionality. The team is further fortifying this separation by implementing Object Relational Mapping (ORM) with the SQLAlchemy toolkit[2].

4.3. API Server and Event Server

The MPO API server exposes its services via RESTful API. The basic entities in the MPO data model are represented with corresponding RESTful resources [29]. The API server utilizes the Model View Controller (MVC) design pattern, and it has been constructed using Flask which is a lightweight micro web application framework written in Python.

The MPO event server is an additional service that runs side-by-side with the API server. It is implemented by utilizing the MDSplus [24] system's event features and provides asynchronous events for real-time updates to clients.

5. Advanced Features in MPO

Now that we have described the basic components, we next describe the more advanced features that support analysis and exploration of workflows and ontology. The Interactive UI Server supports these features and will be described in some detail first before we describe the recent development in supporting ontology evolution and filtering. The functions for supporting instrumentation of user programs are described in detail in earlier publications [7, 9].

5.1. Interactive UI Server

The MPO interactive UI server provides visualization and interactive browsing of the MPO data via the web browser interface. This interface describes and links MPO data and their relationships, while focusing on 3 main MPO data model entities: Workflow, Data Object, and Collection. Workflow is the key concept in this setting and we will only describe this concept. Information on Data Object is given earlier in Section 3 and additional information about Collections can be found in earlier publications [7, 9].

The main Workflow page displays a list of available workflows and their metadata: work-

---

[2] SQLAlchemy is available at http://www.sqlalchemy.org/

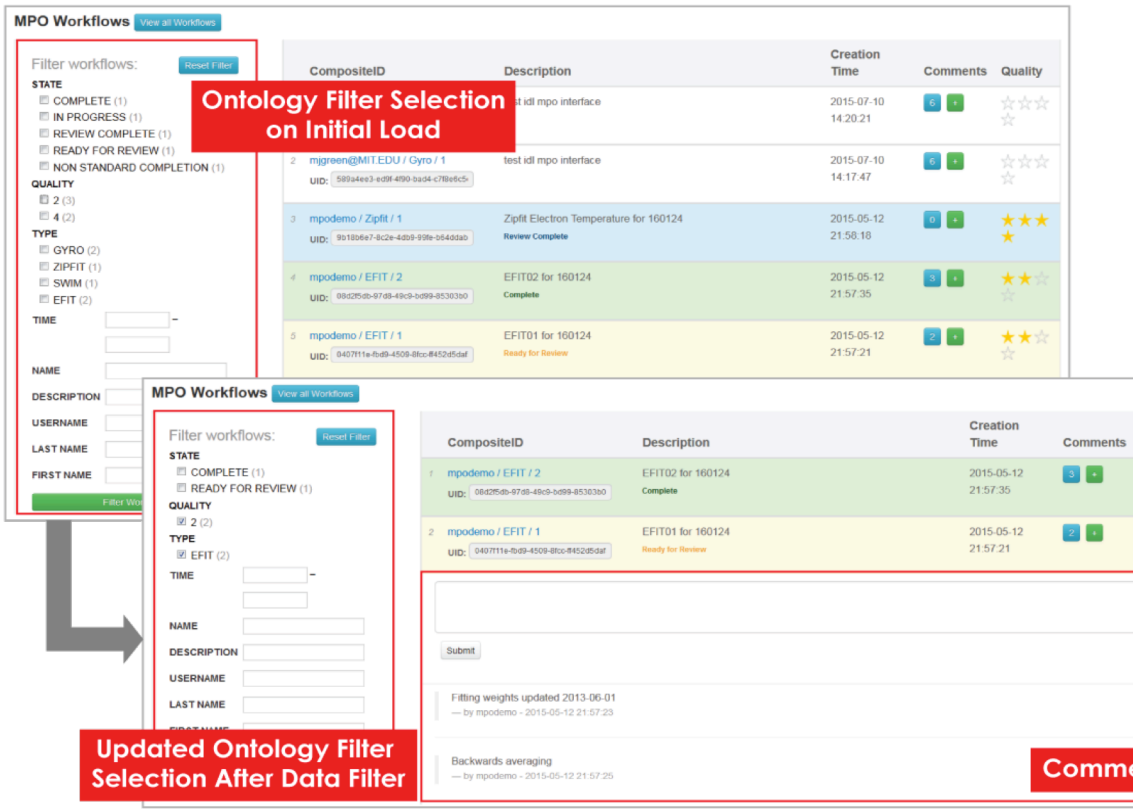

Figure 3: A MPO workflow page partially overlapped with a subset of workflows filtered by conditions on Quality and Type.

flow ID, description, creation time, number of comments and quality ratings. This page also allows users to search and filter through workflows with conditions on metadata and ontology. The support for effective filtering of workflows for interactive exploration has been significantly expanded in the recent months.

The searchable fields are based on general workflow metadata descriptors including creation time range, workflow name, description, the author's username, last name and first name. The ontology fields are based on the user-defined ontology terms. These are dynamic and depend on the resulting list of workflows. Unlike the metadata fields, not all ontology terms are assigned to all workflows, so only the terms used in the displayed workflows will be available for the ontology filter selection. The UI presents these fields in an intuitive and organized manner by grouping them by their parent terms and presenting each field with a counter of associated workflows from the displayed result. Users can click to toggle the checkboxes next to the desired ontology term(s) to filter and narrow down or widen the result set. Figure 5 includes the initial list of all available workflows with the full ontology filter options and an example of a filtered list of workflows with a subset of filter options only used in the filtered list.

5.2. Interactive Workflow Visualization

In the Workflow page, clicking on any of the workflows opens the related Workflow details page. This page provides a rich interactive environment for viewing and exploring all the metadata about the specific workflow, thus creating a dynamic "notebook" interface, see Figure 4 for an example. It utilizes an interactive and real-time graphical interface to present the data associated with a specific workflow. The interface is comprised of an inter-

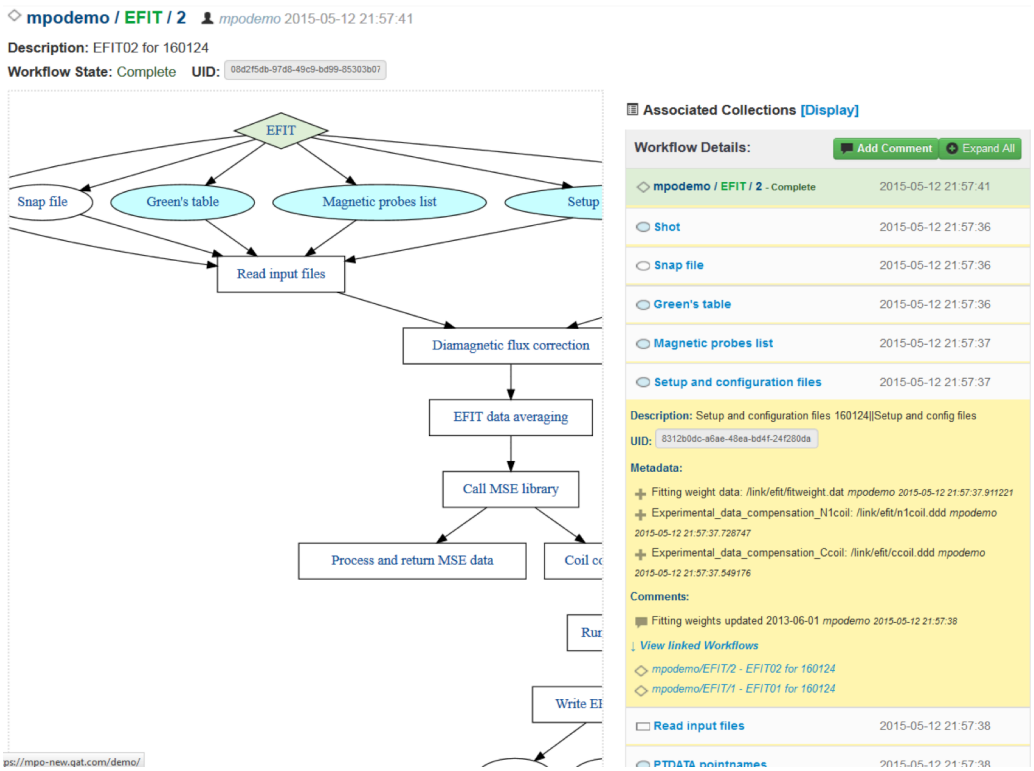

Figure 4: an example of workflow details page, with a workflow from EFIT.

active workflow diagram, an expandable list of nodes that include user comment log, a list of metadata and other linked workflows. The diagram offers pan and zoom capabilities, and also displays metadata associated with the node when selected. The workflow node listing provides users with a chronological list of workflow nodes. When a node is clicked, it expands and reveals all associated metadata.

Users also have the ability to view and add node-specific comments on this workflow page. Using an event system, these respective workflow elements are updated in real-time as new data is added. For example, when a new workflow node is added, the node will appear in the workflow diagram and is also appended to the node list with its metadata. In the DAG visualization, if a workflow element is shared with other workflows, it is automatically detected and highlighted in blue. Figure 4 shows a workflow details page of an EFIT [30] workflow.

5.3. Ontology Evolution

Large collaborations such as those in high-energy physics and fusion could last for a number of decades. Workflows developed in the early years of these projects are gradually evolved to adapt to the new science objectives, new computing hardware and new software infrastructure. In this process, not only the steps of workflows change, but also the terminology and ontology. It is important to support the evolution of terminology and ontology. Our support for ontology evolution takes a practical route, where changes that are more likely to appear in the real applications are supported first.

Generally, ontology contains a hierarchy of categories. Typically, one considers only the categories in the ontology tree, where a higher-level category contains a number of lower-level categories and a leaf node contains only specific values. Some of these values may be arbitrary integers or floating-point values, while others can only be taken from a subset

of string, integers, or floating-point values. In the later case, we say the values of a category are specified by the definition of ontology itself. Each individual value in this case is known as an instance of the category or a term.

Based on our interactions with application scientists, we observe that the most likely change to ontology in physical science is the addition of some terms. This is typically created by the introduction of a new experimental device, a new technique for data collection and analysis, or a new approach to study some physical phenomenon. Our initial attempt at supporting ontology evolution is therefore to add ontology instances without modifying the structure of the ontology tree. The complementary function for addition is removal. Work is planned to support adding and removing of categories, renaming, and modifying categories. These functions will modify the ontology tree. Propagating these changes in the metadata captured will be challenging.

6. Case Studies

The initial development of MPO was tested against a set of experimental data analysis workflows from fusion experiments. In this section, we briefly review an example workflow for EFIT (plasma shape analysis) and then describe our experience of applying MPO to a climate data analysis workflow from a project known as CASCADE.

6.1. EFIT Workflow

EFIT (equilibrium fitting) is a simulation tool for calculating the Magneto-HydroDynamic (MHD) equilibrium in a toroidal magnetically confined plasma[3]. This workflow is instrumented via the MDSplus data acquisition and data management system [31]. Figure 4 has a portion of an instance of the EFIT workflow. This segment shows the main trunk of the workflow, which takes a few input files, connects to the source data (from diagnostic measurements) to extract the necessary parameters for the simulation. The MPO development team has worked with the workflow developers to ensure that only the high level information is captured, which ensures the information presented is a clean workflow diagram, and all interesting aspect of the workflow could be reflected in the figure.

In addition to EFIT, the MPO team has instrumented a number of others including the Simulation of RF Wave Interactions with Magnetohydrodynamics (SWIM), the GYRO turbulence code and some of the codes of the Advanced Tokamak Modeling (AToM) project. The initial results were well received by the OMFIT developers.

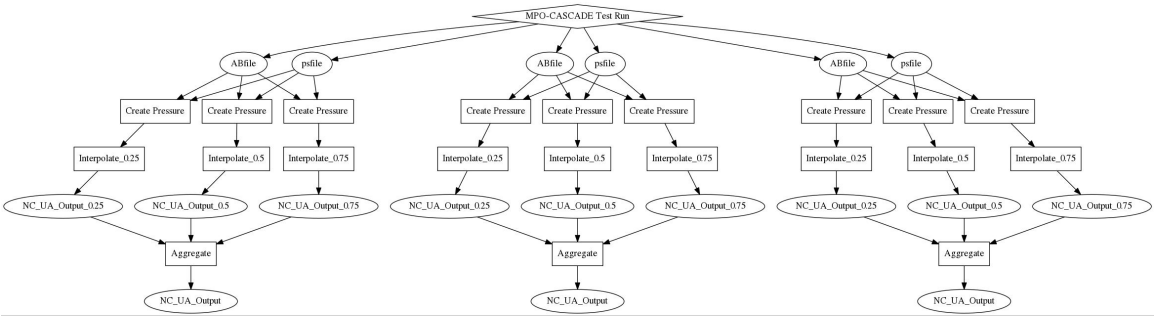

Figure 5: An example workflow from a climate workflow from a project known as CASCADE.

[3] More information about DIII-D is available from < https://fusion.gat.com/global/DIII-D>.

6.2. CASCADE Workflow

To exercise the functionality of MPO, we also apply it to a set of workflows from a climate data analysis project known as CASCADE [32, 33]. This particular workflow is a simplified version for detecting Atmospheric Rivers [32, 33], a process is primarily a data parallel in nature. The particular run of the workflow starts with hundreds of data files, which creates hundreds of identical branches of the DAG, where each of the branches is essentially the same. To reduce clutter, we have chosen to only show three such branches in Figure 5.

This is one of the first workflow instrumented without hands-on help from the MPO developer team. The graduate student who performed the instrumentation spent only a few hours on the task and was able to instrument about half a dozen functions. The test run was conducted on a supercomputer named Edison at National Energy Research Scientific Computing (NERSC) center located at Berkeley California. From the captured metadata, we created a simple high-level flow diagram that shows the essence of the workflow. Though this test went smoothly, it does reveal one shortcoming of the existing system; it produced too many copies of the identical subtrees. A better ways to represent them in a more compact way would be highly desirable.

6.3. Lessons Learned

From the implementation and testing of MPO, we also learnt a few lessons. For example, an initial implementation of the filtering function was taking a long time when there are thousands of workflows in MPO database. To reduce the response time, we have to implement more advanced techniques to process the filters. In a number of tests, we found that the DAG produced by the workflows is too deep or too wide to visualize. For example, the CASCADE workflow has too many nearly identical subtrees and the workflow from SWIM has too many levels of details. The Container is one way we plan to deal with the many nearly identical subtrees. To deal with deep hierarchies, we are in a process to develop a way to collapse a subtree into a supernode.

7. Conclusions and Future Work

The MPO system automates documentation of scientific workflows and associated information and does so independently of the workflow orchestration mechanism. It provides capabilities for organizing and analyzing documented metadata as well as presenting it with interactive visual representations.

In the near term, the team is working wrapping the current development version into a public release. Additionally, we are reaching out to more users to exercise the MPO system and demonstrate its usefulness for scientific applications. One specific application we are working on as of this writing is a tomography reconstruction workflow. Like many such workflows, this one has also gone through decade-long organic evolution to produce a tangle of complex scripts. MPO would be able to help the user to clearly document the key steps of the workflow and the interaction of its many components.

The early experience demonstrates that the MPO system and design are not only useful for individual researchers to document their own computations, but also for an environment for collaboration among team members. Collaborating scientists can easily see what data and computational codes are being shared between them, or what kind of workflows others are running to solve a similar problem. Therefore, providing tools and features for promoting collaborations in the immediate future can be useful. Some examples are “subscribe” feature for getting notification on somebody else’s workflow, and “like” feature for highlighting some high-value Data Object or Workflows.

Another potential area for enhancement is providing data exchange capabilities with relevant provenance tools and workflow engines. The World Wide Web Consortium (W3C) published a set of provenance standards and recommendations named PROV. Multiple supporting applications exist and some of them are complimentary to MPO capabilities. The MPO team plans to provide an export mechanism so that data collected by the MPO System can be used by PROV standard supported software.

8. Acknowledgement

This work was supported by the US DOE, Office of Advanced Scientific Computing Research and the Office of Fusion Energy Sciences under DE-SC0008697, DEAC02-05CH11231, and DE-SC0008736.

9. References

1. Simmhan, Y.L., B. Plale, and D. Gannon, *A survey of data provenance in e-science.* SIGMOD Rec., 2005. **34**(3): p. 31-36.
2. Gruber, T.R., *A translation approach to portable ontology specifications.* Knowledge Acquisition, 1993. **5**(2): p. 199-220.
3. Noy, N.F. and D.L. McGuinness, *Ontology development 101: A guide to creating your first ontology*. 2001.
4. Altintas, I., O. Barney, and E. Jaeger-Frank, *Provenance collection support in the kepler scientific workflow system*, in *Provenance and annotation of data*. 2006, Springer. p. 118-132.
5. Davidson, S.B., et al., *Provenance in Scientific Workflow Systems.* IEEE Data Eng. Bull., 2007. **30**(4): p. 44-50.
6. Schissel, D.P., et al., *Automated metadata, provenance cataloging and navigable interfaces: Ensuring the usefulness of extreme-scale data.* Fusion Engineering and Design, 2014. **89**(5): p. 745-749.
7. Wright, J.C., et al., *The MPO API: A tool for recording scientific workflows.* Fusion Engineering and Design, 2014. **89**(5): p. 754-757.
8. Greenwald, M., et al., *A metadata catalog for organization and systemization of fusion simulation data.* Fusion Engineering and Design, 2012. **87**(12): p. 2205-2208.
9. Abla, G., et al., *The MPO System for Automatic Workflow Documentation*. 2015.
10. Myers, J.D., E.S. Mendoza, and B. Hoopes. *A Collaborative Electronic Laboratory Notebook*. in *IMSA*. 2001. Citeseer.
11. Erickson, J. and D. Eppstein, *Iterated nearest neighbors and finding minimal polytopes.* Discrete and Computational Geometry, 1994. **11**: p. 321-350.
12. Inmon, W.H., B. O'Neil, and L. Fryman, *Business Metadata: Capturing Enterprise Knowledge: Capturing Enterprise Knowledge*. 2010: Morgan Kaufmann.
13. Bavoil, L., et al. *Vistrails: Enabling interactive multiple-view visualizations*. in *Visualization, 2005. VIS 05. IEEE*. 2005. IEEE.
14. Pordes, R., et al. *The open science grid*. in *Journal of Physics: Conference Series*. 2007. IOP Publishing.
15. Altunay, M., et al., *A science driven production Cyberinfrastructure—the Open Science grid.* Journal of Grid Computing, 2011. **9**(2): p. 201-218.
16. Singh, G., et al. *A Metadata Catalog Service for Data Intensive Applications*. in *SC03*. 2003.
17. Deelman, E., et al. *Grid-Based Metadata Services*. in *the 16th International Conference on Scientific and Statistical Database Management,*. 2004.
18. Stojanovic, L., et al., *User-driven ontology evolution management*, in *Knowledge engineering and knowledge management: ontologies and the semantic web*. 2002, Springer. p. 285-300.

19. Noy, N.F. and M. Klein, *Ontology evolution: Not the same as schema evolution.* Knowledge and information systems, 2004. **6**(4): p. 428-440.
20. Kondylakis, H. and D. Plexousakis, *Ontology evolution without tears.* Web Semantics: Science, Services and Agents on the World Wide Web, 2013. **19**: p. 42-58.
21. Zablith, F., et al., *Ontology evolution: a process-centric survey.* The Knowledge Engineering Review, 2015. **30**(01): p. 45-75.
22. Freire, J., et al., *Provenance for Computational Tasks: A Survey.* Computing in Science & Engineering, 2008. **10**(3): p. 11-21.
23. Scheidegger, C., et al., *Querying and creating visualizations by analogy.* IEEE Transactions on Visualization & Computer Graphics, 2007(6): p. 1560-1567.
24. Stillerman, J., et al., *MDSplus data acquisition system.* Review of Scientific Instruments, 1997. **68**(1): p. 939-942.
25. Folk, M., et al. *An overview of the HDF5 technology suite and its applications*. in *Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases*. 2011. ACM.
26. Crockford, D., *The application/json media type for javascript object notation (json)*. 2006, IETF.
27. Shafranovich, Y., *Common format and MIME type for Comma-Separated Values (CSV) files*. 2005, IETF.
28. Richardson, L. and S. Ruby, *RESTful web services*. 2008: O'Reilly Media, Inc..
29. Fielding, R.T. and R.N. Taylor, *Principled design of the modern Web architecture.* ACM Trans. Internet Technol., 2002. **2**(2): p. 115-150.
30. Fellinger, P., et al., *Numerical modeling of elastic wave propagation and scattering with EFIT—elastodynamic finite integration technique.* Wave motion, 1995. **21**(1): p. 47-66.
31. Lao, L., et al., *Reconstruction of current profile parameters and plasma shapes in tokamaks.* Nuclear fusion, 1985. **25**(11): p. 1611.
32. Byna, S., et al. *Detecting atmospheric rivers in large climate datasets*. in *the 2nd international workshop on Petascal data analytics: challenges and opportunities (PDAC'11)*. 2011.
33. Jeon, S., et al., *Characterization of extreme precipitation within atmospheric river events over California.* Advances in Statistical Climatology, Meteorology and Oceanography, 2015. **1**(1): p. 45.