The International Journal of Digital Curation
Volume 7, Issue 1 | 2012

Golden Trail: Retrieving the Data History that Matters from a Comprehensive Provenance Repository

Paolo Missier,
Newcastle University

Bertram Ludäscher, Saumen Dey, Michael Wang and Tim McPhillips,
UC Davis, University of California

Shawn Bowers and Michael Agun,
Gonzaga University

Ilkay Altinas,
UC San Diego, University of California

Abstract

Experimental science can be thought of as the exploration of a large research space, in search of a few valuable results. While it is this "Golden Data" that gets published, the history of the exploration is often as valuable to the scientists as some of its outcomes. We envision an e-research infrastructure that is capable of systematically and automatically recording such history – an assumption that holds today for a number of workflow management systems routinely used in e-science. In keeping with our gold rush metaphor, the provenance of a valuable result is a "Golden Trail". Logically, this represents a detailed account of how the Golden Data was arrived at, and technically it is a sub-graph in the much larger graph of provenance traces that collectively tell the story of the entire research (or of some of it).

In this paper we describe a model and architecture for a repository dedicated to storing provenance traces and selectively retrieving Golden Trails from it. As traces from multiple experiments over long periods of time are accommodated, the trails may be sub-graphs of one trace, or they may be the logical representation of a virtual experiment obtained by joining together traces that share common data.

The project has been carried out within the Provenance Working Group of the Data Observation Network for Earth (DataONE) NSF project. Ultimately, our longer-term plan is to integrate the provenance repository into the data preservation architecture currently being developed by DataONE.

 doi:10.2218/ijdc.v7i1.221

Introduction

Experimental science is not a linear process. As we have noted in prior work (Altintas et al., 2010), publishable results routinely emerge at the end of an extended exploratory process, which unfolds over time and may involve multiple collaborators, who often interact only through data sharing facilities. This is particularly apparent in e-science, where experiments are embodied by computational processes which can be executed repeatedly and in many parametric variations, over a large number of input configurations. These processes typically encompass a combination of well-defined specifications encoded as scientific workflows,for example, in scientific workflow environments like Kepler (Ludäscher et al., 2006) or Taverna (Turi et al., 2007), or as custom-made scripts to move data across repositories, to execute scientific codes on remote supercomputers, and so on.

Regardless of the specific computational model chosen, current implementations of e-science infrastructure are primarily designed to support the discovery and creation of valuable data outcomes, while result dissemination and a description of how these results were achieved have been largely confined to the "materials and methods" sections in traditional research paper publications. Spurred, in part, by pressure from funding bodies, which are interested in maximizing their return on investment, the focus of e-science research is now shifting onto the later phases of the scientific data lifecycle: namely the sharing and dissemination of scientific results, with the key requirements that the experiment be repeatable, and the results be *verifiable* and *reusable* (Nature, 2009). The notion of *Research Objects* (RO) is emerging in response to these needs (Bechhofer et al., 2011). These are bundles of logically related artefacts that collectively encompass the history of a scientific outcome and can be used to support its validation and reproduction. They may include the description of the processes used, i.e., workflows, along with the *provenance traces* obtained during the execution of these processes. Additionally, multiple executions may be chained together by one or more scientists in an exploratory fashion, resulting in multiple paths of trials and errors until successful outcomes with scientific value are achieved.

Importantly, ROs provide a view of the experimental process that is focused on a selected few datasets that are destined for publication, rather than on the entire "raw" exploration. As a result, such a view is a "virtual" one, in the sense that it represents a linear and uniform account of the research, obtained by sifting through a potentially large space of partial and possibly unrelated, insignificant or invalid intermediate results, which were generated at different times, possibly by multiple collaborators who operate using different e-research environments.

The project described in this paper stems from the observation that, despite such heterogeneity of tools and programming models, *experiment virtualization* is still possible on two main conditions: that the repositories used by participants to share their data can map different identifiers used to reference the same datasets; and that the provenance traces captured by different e-infrastructures can be mapped to a common provenance data model. We have used these assumptions in our recent *Data Tree of Life* project (Missier et al., 2010), where we have shown how multiple, independently produced provenance traces expressed using the Open Provenance Model (OPM) can be successfully "joined up" when they share references to data

items that have been deposited in provenance-aware data repositories. In general, this step cannot always be completely automated and requires an explicit curation step with the scientist's direct involvement. The resulting composite trace effectively represents evidence of a virtual experiment, in which the outcome of one process has been uploaded to a repository, and later independently used as an input to another process.

The project described in this paper is a logical continuation of that effort. Here we focus on a scenario where scientists explore an experimental space through repeated execution of a variety of workflows. Each execution generates a provenance trace, and all the traces are stored in a shared provenance repository. We have termed the project *Golden-Trail*, to emphasize that the repository architecture enables scientists to generate a "clean" account of their most valuable findings (the "golden data"), out of many possible, often only exploratory, analysis paths. This short project is part of the much larger Data Observation Network for Earth (DataONE) project,[1] one of several Data Conservancy projects funded by the NSF over the past few years. Ultimately, our plan is to integrate the provenance repository into the DataONE data preservation architecture.

In the rest of the paper we discuss the challenges associated with the main elements of the repository model and architecture:

- A *provenance model* for describing the lineage of process-generated data. The model combines the core data dependencies that are part of the Open Provenance Model (OPM) (Moreau et al., 2011) with a description of the process that generated the data. This enables us to provide an explicit representation of the workflow structure, along with a correspondence between its elements and those of the provenance trace. Making such correspondence explicit in the model results in a more natural and intuitive provenance query and presentation model. We plan to evolve our generic schema for representing workflows in order to accommodate the most common workflow models that are in broad use in e-science, including Kepler, Taverna, VisTrails (Callahan et al., 2006), Pegasus (Kim et al., 2008), Galaxy (Nekrutenko, 2010), and eScience Central (Hiden et al., 2011). We denote our model D-OPM, to indicate that it is a backward-compatible extension of the OPM;
- A *provenance repository* for storing the "raw" provenance traces obtained from multiple executions of one or more processes, which represent the actual exploratory phase of scientific investigation;
- A *user environment* for the semi-automated construction of virtualized accounts of an experiment. The environment consists of two components: (i) a query interface into the repository, by which the scientist can explore and visualize the space of available traces, guided by the process specification part of D-OPM, and (ii) a *curation interface* by which scientists provide the necessary mappings across data generated by different traces (an explicit data curation step).

[1]DataONE: http://www.dataone.org

 doi:10.2218/ijdc.v7i1.221

Provenance Model

The D-OPM (for DataONE Provenance Model) is a light-weight data model for representing the provenance of data that is generated *through a formalized process*. As mentioned earlier, we initially focus on workflows as a prime example of such process specifications. Our plan is to gradually expand the representation of structured processes beyond workflow, to include scripting languages used in science, such as R.

In every case, data dependency relations are derived from the observation of one execution of the process, in line with the Open Provenance Model (OPM). In the workflow context, these relations specifically represent the production and consumption of data items by workflow elements ("actors"). In addition, however, D-OPM captures an extended provenance trace, which also includes a representation of the structure of the workflow itself.[2] Such an extension provides an important reference context for presenting provenance to users, in much the same way as program debugging information is normally associated with the program's source code. In the next section we show in more detail how the model can be exploited, by presenting a categorization of queries over extended traces.

The provenance model includes the following key elements:

Structural elements:

- **Actor:** a single computational step, and
- **Workflow:** an orchestration of a collection of actors with data and control dependencies. Workflows can be statically or dynamically nested, i.e., an actor can expand into a whole sub-workflow.

Runtime elements:

- A **Run:** representing a single execution of an entire workflow. It consists of Actor **invocations**, i.e., executions of individual steps within the workflow;
- **Data Items:** representing data values[3] that are either produced or consumed by Actor Invocations;
- **Data dependencies:** corresponding to *observable* events, namely generation (*DataGen*) and consumption (*DataUse*) of a data item by an actor invocation;
- The **Attribution** of a run: i.e., a reference to users who run the workflow and thus "own" the traces.

[2] Such a representation of process structure is necessarily out of the scope of the OPM, which is process-agnostic, but can be added to it by means of the *profiles* mechanism.

[3] In this simple model we do not make any distinction between atomic data values, and data structures

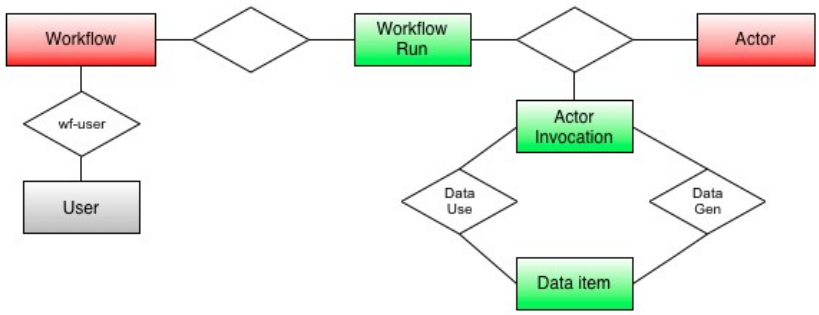


Figure 1. Minimal version of the D-OPM model, implemented in the current prototype.

Provenance Queries

The simple model in Figure 1 is sufficient to illustrate the synergy between the structural portion of the model (Workflow, User, Actor), and the runtime portion (Workflow Run, Actor Invocation, Data Item) along with the core data consumption and generation events. A broad variety of queries are supported by the model. Listed here (expressed using a Datalog-like notation) is a non-exhaustive core set of queries. The derived relations (i.e., views) computed by these queries can then be further composed into more complex queries. Some examples are given below.

Data and actor-level queries

Ancestor queries

1. Find all Actors that directly or indirectly contributed to the generation of data item D (backwards traversal). This is the set of actors that satisfy the query **dep*(D,A)**, where:

$$dep^*(D,A) \leftarrow genBy(D,I), invocation(I,A).$$
$$dep^*(D,A) \leftarrow genBy(D,I'), idep^*(I',I), invocation(I,A).$$
$$idep^*(I,I') \leftarrow idep(I,I').$$
$$idep^*(I,I') \leftarrow idep(I,I''), idep^*(I'',I').$$
$$idep(I,I') \leftarrow used(I,D), genBy(D,I').$$

2. Find all data items D' that directly or indirectly contributed to the generation of data item D. This is the set of D' that satisfy the query **ddep*(D,D')**, where:

$$ddep(D,D') \leftarrow genBy(D,I), used(I, D').$$
$$ddep^*(D,D') \leftarrow ddep(D,D'), ddep^*(D'',D').$$
$$ddep^*(D,D') \leftarrow ddep(D,D').$$

Notice that the same rules can be used to perform a *descendant query*.

 doi:10.2218/ijdc.v7i1.221

Descendant queries

3. Find all data items that directly or indirectly have been affected by data item D (forward traversal). In fact, this is the set D of items that satisfy the query

ddep*(D,D′), given D′.

Workflow-level queries

4. Find all data that flowed through a workflow W during one of its runs:

wf_dep(W,D) ← used(I,D), invocation(I,R), run(R,W).

wf_dep(D,W) ← genBy (D,I), invocation(I,R), run(R,W).

User-related queries

5. Find all data items that a user either used or generated:

user_dep(U,D) ← used(I,D), invocation(I,R), run(R,W), workflow(W,U).

user_dep(D,U) ← genBy (D,I), invocation(I,R), run(R,W), workflow(W,U).

Provenance Repository Application and Architecture

We have implemented a prototype for the *Golden-Trail* provenance repository that is designed to be integrated with the main DataONE architecture.[4]

Golden-Trail Application

The *Golden-Trail* application is built on four logical components: the *User Interface*, the *Trace Parser*, the *Graph Visualization*, and the *Data Store* (see Figure 2).

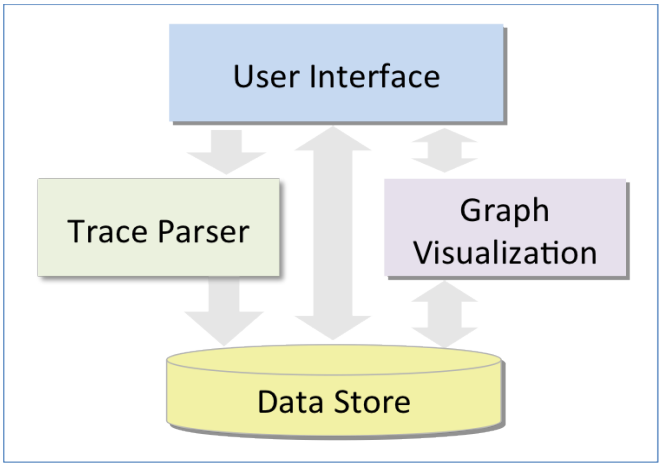


Figure 2. *Golden-Trail* application.

The *User Interface* allows a scientist to interact with the provenance repository. The scientist can upload trace files into the repository and execute queries. Query

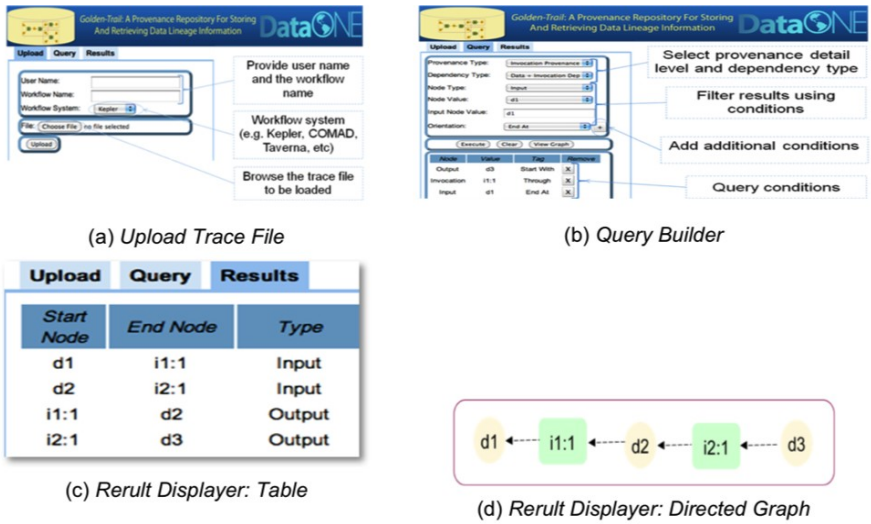
[4]A more complete account of the architecture can be found in a separate technical report, available at http://www.cs.ucdavis.edu/research/tech-reports/2011/CSE-2011-16.pdf

results are displayed as tables and as dependency graphs. The *Trace Parser* parses trace files from different workflow systems, such as Kepler or Taverna. New custom parsers from other workflow systems can easily be added. The *Graph Visualization* component renders a provenance trace as a directed acyclic graph in an interactive manner. The *Data Store* holds provenance traces and provides an API to upload and query provenance data.

The *Golden-Trail User Interface* has three primary functions: *Upload Trace File*, *Query Builder*, and *Result Display*.

The *Upload Trace File* allows a scientist to upload provenance data (a trace file) to the provenance repository. In the upload page, shown in Figure 3a, the scientist provides the user name, the workflow name and the workflow system name. The latter is used to invoke the appropriate trace parser. The scientist then chooses a trace file using the dialog box and initiates the upload process by clicking the upload button. The *Query Builder* (Figure 3b) can be used to interactively specify queries against the provenance repository. This is done by selecting (i) a provenance view, (ii) a dependency view, and (iii) a set of query conditions.

The provenance view is used to define the desired abstraction level at which results are to be returned. Provenance traces can be abstracted at the user, workflow, run, actor, and invocation levels. For example, users who only care about the run level may not want to view the details of individual invocations. After selecting an abstraction level, the dependency view needs to be defined, namely as a *data dependency graph* (i.e., how a data item depends on other data items), an *invocation dependency graph* (i.e., how an invocation depends on other invocations), or a combination of the two. Finally, a set of query conditions can be specified, using a set of starting nodes (data items or invocations), intermediate nodes, and end nodes.



(a) *Upload Trace File* (b) *Query Builder*

(c) *Rerult Displayer: Table* (d) *Rerult Displayer: Directed Graph*

Figure 2: *Golden-Trail Application*

Figure 3. *Golden-Trail* application user interface.

 doi:10.2218/ijdc.v7i1.221

After a query is executed, *Golden-Trail* renders the result in two different formats: (i) as a table with a dependency presented as a row, specifying that the "End Node" is dependent on the "Start Node" as shown in Figure 3c; or (ii) as a dependency graph displaying the dependencies from a right node (data item or invocation) to a left node, as shown in Figure 3d.

The *Trace Parser* handles trace files coming from specific workflow systems for which a parser is available. It makes the provenance data from the trace file D-OPM compatible and loads it into the provenance repository. Some workflow systems share data items (i.e., one workflow run generates a data item and another workflow run uses that data item). In case two workflow runs maintain the same data identifiers of a shared data item, the *Trace Parser* links their respective gen-by/used relations based on the shared data identifier, (i.e., by *stitching* two provenance graphs to form a larger graph). The *Golden-Trail Graph Visualization* renders a query result as a dependency graph, in addition to the tabular format. The result can be displayed either as an interactive or a static dependency graph (i.e., as an image). Interactive graphs can be incrementally expanded.

Golden-Trail Architecture

The *Golden-Trail* is developed using the GWT (Google Web Toolkit) framework and built on the three-tier J2EE architecture. The client-side code (*Upload GUI*, *Query GUI*, and *GWT client-server interface*) resides in the web server. The server-side code (*Upload Trace File*, *Query Builder*, and *Result Displayer*) resides in the application server. Tomcat is used to serve as both the web server and the application server for this prototype development. The final tier is our database server. The overall interactions of all the components are shown in Figure 4.

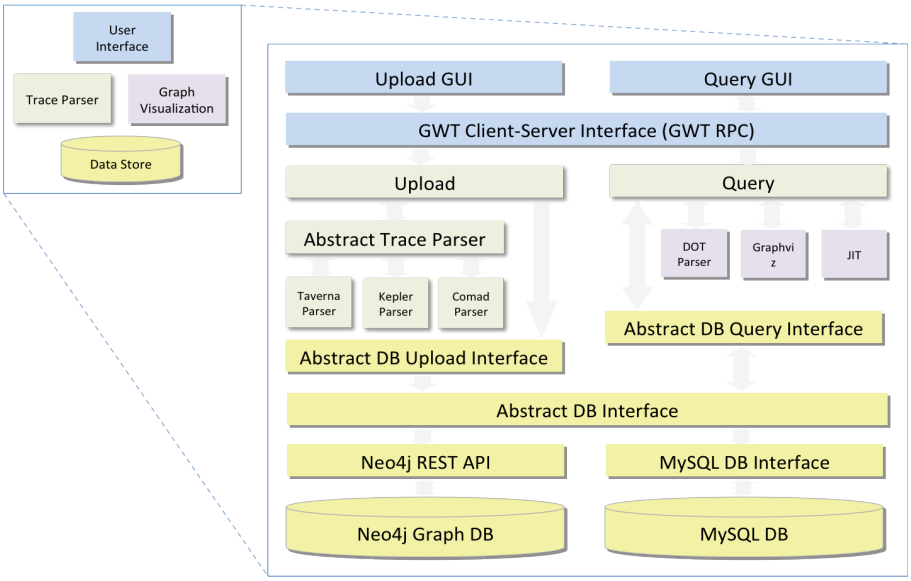


Figure 4. *Golden-Trail* architecture.

The GWT Client-Server Interface makes asynchronous calls to the respective server-side components. The server-side *Upload* component is invoked in case of an *Upload Trace File* request. The *Upload* component calls an appropriate *Trace Parser*

(based on the selection of the workflow system). The *Trace Parser* parses the trace file and creates a provenance model object, which is passed to the *Abstract DB Upload Interface*. This *DB Upload Interface* prepares a set of DML statements for the targeted database and calls the respective database server API.

Golden-Trail provides an extensible database layer, which is implemented using the abstract factory design pattern. Currently, it supports a relational database and a graph database. In the relational database, the provenance model is implemented using a set of tables and relationships. In the graph database, the provenance model is implemented as a graph with a set of nodes. Each of these two models specializes an abstract graph model consisting of generic-type nodes and relationships amongst nodes (*used* and *gen-by* dependencies are examples of specializations).

We have implemented a relational database (MySQL) and a graph database (Neo4j) as the data servers for the *Golden-Trail* prototype. A typical provenance query is recursive in nature. Executing such queries in Neo4j is relatively easy, as it provides a set of REST APIs for querying with recursions. We used these features in *Golden-Trail*. MySQL does not provide such constructs. We developed a set of stored procedures to achieve the recursion.

Experimental Testbed

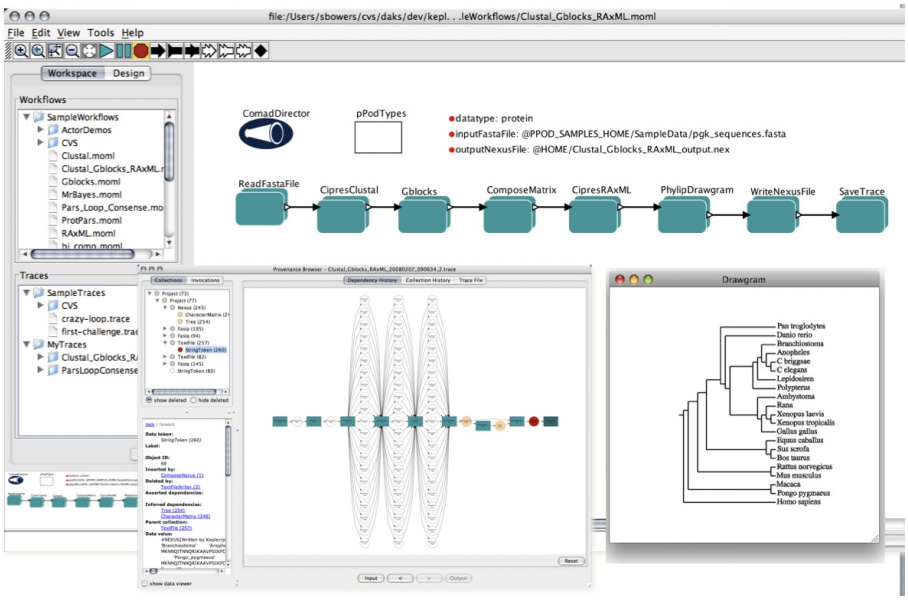


Figure 5. Phylogenetics workflow (top) with provenance trace (bottom) from the Kepler/pPOD package using the COMAD module.

Our experimental testbed consists of a suite of pre-existing Kepler workflows, prepared from the "Tree of Life"/pPOD project (Bowers et al., 2008). The pPOD testbed includes a suite of workflows for performing various phylogenetic analyses using a library of reusable components for aligning biological sequences, and inferring phylogenetic trees based on molecular and morphological data. The workflows are divided into various subtasks that can be run independently as smaller, exploratory workflows for testing different parameters and algorithms, or combined into larger

 doi:10.2218/ijdc.v7i1.221

workflows for automating multiple data access, tree inference, and visualization steps. A number of the smaller workflows within pPOD are designed explicitly to be run over output generated from other workflows within the suite.

Having demonstrated provenance interoperability and integration as part of a previous effort (Missier et al., 2010), the emphasis has been less on experimenting with specific provenance integration techniques. Instead, we focused on populating the repository using multiple executions of multiple workflow fragments, each related to each other through their input and output (sometimes intermediate) data products, and on testing query functionality to extract Golden-Trails from the repository. More specifically, we demonstrate query capability with different views of the results, including returning and rendering all or a portion of a run graph where nodes represent whole workflow runs, and possibly with data nodes as intermediate connections, as the result of a query, emphasizing the lineage of data across different e-science infrastructures.

To demonstrate all the query capabilities, we developed the following synthetic experiment involving three workflows. Two scientists (user1 and user2) participated in this experiment. The dependencies among the workflows are as shown in Figure 6. The first workflow (wf1) was executed first, then the second (wf2) and third (wf3) workflows used output data items from wf1's execution.

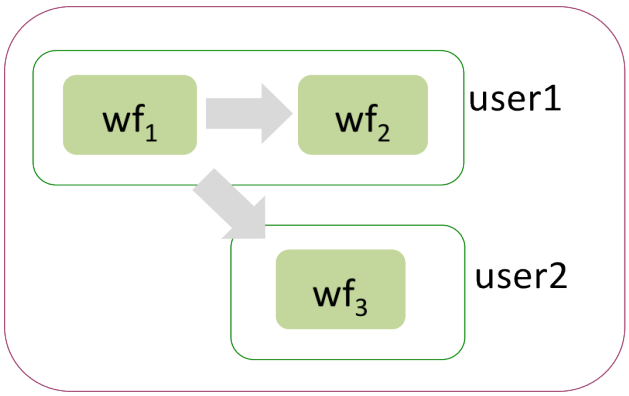


Figure 6. Workflows wf2 and wf3 use data items from wf1.

During the execution of each of these workflows, the respective workflow systems capture processing histories in trace files. Many of the existing systems can capture invocations, which are instances of a process or actor. Others can only capture general input/output dependencies. Our system handles both types of provenance traces.

In case two trace files use the same identifier for shared data items, *Golden-Trail* can use the techniques developed for the DataTree of Life project (Altintas et al., 2010) to stitch them automatically. In our synthetic experiment, workflows wf2 and wf3 use data items from workflow wf1 and use common identifiers for the shared data items. Thus, after loading all three trace files, all three provenance graphs can be stitched together to produce the provenance graph of the entire experiment. After all trace files are loaded into the *Golden-Trail* repository, they can be queried as indicated earlier.

Conclusions

In our prior recent work (Altintas et al., 2010), we began an investigation around the concept of a *virtual experiment*, that is, a unified representation of multiple scientific experiments, which are logically connected through shared data. The key condition for building such unified representations is that a provenance trace for each of the individual experiments be available in some agreed-upon format. In this paper we have described a model and architecture for a provenance repository, out of which virtual experiment views can be extracted. We have assumed, for simplicity, that experiments are carried out using workflows, and that each execution generates a provenance trace. The traces may be generated by multiple systems, but are mapped to our common repository model, D-OPM. We have described the simplified version of the model that we have implemented as part of the *Golden-Trail* project, and a prototype architecture for the repository, with upload and query capabilities.

The project has been carried out within the Provenance Working Group of the Data Observation Network for Earth (DataONE) NSF project. Ultimately, our plan is to integrate the provenance repository into the data preservation architecture currently being developed by DataONE.

Acknowledgements

We gratefully acknowledge the NSF DataONE project that made this project possible by funding two interns during the summer of 2011.

References

Altintas, I., Anand, M. K., Ludaescher, B, Bowers, S., Crawl, D., Belloum, A., Missier, P., Goble, C., & Sloot, P. (2010). Understanding collaborative studies through interoperable workflow provenance. Paper presented at the International Provenance And Annotation Workshop, Troy, NY.

Bechhofer, S., Buchan, I., De Roure, D., Missier, P., & Al., E. (2011). Why linked data is not enough for scientists. *Future Generation Computer Systems (FGCS) Journal*. doi:10.1016/j.future.2011.08.004

Bowers, S., McPhillips, T.M., Riddle, S., Anand, M.K., & Ludäscher, B. (2008). Kepler/pPOD: Scientific workflow and provenance support for assembling the tree of life. Paper presented at the International Provenance And Annotation Workshop,Troy, NY.

Callahan, S.P., Freire, J., Santos, E., Scheidegger, C.E., Silva, C.T., & Vo, H.T. (2006). VisTrails: Visualization meets data management. In proceedings of the 2006 ACM SIGMOD International Conference on Management of Data. doi:10.1145/1142473.1142574

Hiden, H., Watson, P., Woodman, S., & Leahy, D. (2011). *e-Science Central: Cloud-based e-Science and its application to chemical property modelling.* Technical Report CS-TR-122, School of Computing Science, Newcastle University.

Kim, J., Deelman, E., Gil, Y., Mehta, G., & Ratnakar, V. (2008). Provenance trails in the Wings-Pegasus system. *Concurrency and Computation: Practice and Experience, 20*(5). doi:10.1002/cpe.1228

Ludäscher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M., Lee, E.A., et al. (2006). Scientific workflow management and the Kepler system. *Concurrency and Computation: Practice and Experience, 18*(10). http://dx.doi.org/10.1002/cpe.994

Missier, P., Ludaescher, B., Bowers, S., Anand, M. K., Altintas, I., Dey, S., & Sarkar, A. (2010). Linking multiple workflow provenance traces for interoperable collaborative science. Paper presented at the 5th Workshop on Workflows in Support of Large-Scale Science (WORKS), Seattle, Washington.

Moreau, L., Clifford, B., Freire, J., Futrelle, J., Gil, Y., Groth, P., Kwasnikowska, N., et al. (2011). The Open Provenance Model: Core specification (v1.1). *Future Generation Computer Systems, 7*(21). http://dx.doi.org/10.1016/j.future.2010.07.005

Nature. (2009). Special Issue on Data Sharing. *Nature 461*.

Nekrutenko, A. (2010). Galaxy: A comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biology, 11*(8). doi:10.1186/gb-2010-11-8-r86

Turi, D., Missier, P., Roure, D.D., Goble, C., & Oinn, T. (2007). Taverna workflows: Syntax and semantics. Paper presented at the third e-Science conference. Bangalore, India. doi:10.1109/E-SCIENCE.2007.71