Tracking Provenance in ORNL's Flexible Research Platforms

Zachary Hensley (zphensley42@students.tntech.edu),[1] Jibonananda Sanyal (sanyalj@ornl.gov),[2] and Joshua New (newjr@ornl.gov)[2]

[1)] *Tennessee Technological University*

[2)] *Oak Ridge National Laboratory (Energy and Transportation Sciences Division)*

Provenance is defined as information about the origin of objects, a concept that applies to both physical and digital objects and often overlaps both. The use of provenance in systems designed for research is an important but forgotten feature. Provenance allows for proper and exact tracking of information, its use, its lineage, its derivations and other metadata that are important for correctly adhering to the scientific method. In our project's prescribed use of provenance, researchers can determine detailed information about the use of sensor data in their experiments on ORNL's Flexible Research Platforms (FRPs). Our project's provenance system, Provenance Data Management System (ProvDMS), tracks information starting with the creation of information by an FRP sensor. The system determines station information, sensor information, and sensor channel information. The system allows researchers to derive generations of experiments from the sensor data and tracks their hierarchical flow. Key points can be seen in the history of the information as part of the information's workflow. The concept of provenance and its usage in science is relatively new and while used in other cases around the world, our project's provenance differs in a key area. To keep track of provenance, most systems must be designed or redesigned around the new provenance system. Our system is designed as a cohesive but separate entity and allows for researchers to continue using their own methods of analysis without being constrained in their ways in order to track the provenance. We have designed ProvDMS using a lightweight provenance library, Core Provenance Library (CPL).[6] In addition to keeping track of sensor data experiments and its provenance, ProvDMS also provides a web-enabled visualization of the inheritance.

I. INTRODUCTION

Provenance, defined as the origin of objects, is important in tracking data used in research. Provenance systems involve a level of overhead in that they must store extra meta-information in addition to the original data. Though this can be an issue, the most important aspect when designing a provenance system is how to effectively track the usage of the data. Our system, ProvDMS, handles this difficulty without impeding the workflows of our users — maintaining independence via a non-restrictive environment. This approach allows us to demonstrate the effectiveness of our system for use with the Flexible Research Platforms (FRPs).

Our system uses a combination of PHP and JavaScript to retrieve information from an LNDB server (LoggerNet Database).[3] The information from the FRPs is automatically logged to the LNDB server in a structured format. POST data can be modified to allow any specification of stored information to be used with our provenance system; however, the current setup is defined to specifically use MySQL. More information regarding the structure of our system is located in *Section 4 - Design*.

II. STRUCTURE AND PURPOSE

Provenance systems allow for users to track the use of specific information. The use of these systems are often very specific to the needs of the users and the structure of the data. The initial challenges in designing our system was the layout and structure of the data to be tracked and which information needed to be accessible for the users.

A. Flexible Research Platforms (FRPs)

The data our system is designed for comes from platforms known as FRPs. These platforms are a result of the *Maximum Building Energy Efficiency Research Laboratory* (MAXLAB) project and are heavily instrumented with hundreds of channels of data at 30-second resolution.[1] With current data acquisition system hardware, we can support up to a total number of physical sensor channels of 2028 (676 in 1-story, 1352 in 2 story). These sensor channels are capable of logging information at many resolutions and give a good picture of the buildings' long term energy usage.

One of the issues with using information from the FRPs in experiments involves the process of deriving new information from logged information. Once an experiment completes its life-cycle, it is likely that the information originally from the FRPs has been modified numerous times. It is also very likely that the information has been updated during the experiment, with newer information from the FRPs being used for the experiment. These issues highlight the purpose of the provenance system.

III. RESTRICTIONS

A. Nature of provenance systems

Many current provenance systems focus on specific restrictions for the users in order to provide proper and accurate provenance. Adriane Chapman et al[4] spoke of a list desiderata ("desired things") for future provenance systems to maintain. Their list mentions many aspects we focused on, including provenance granularity, user interactions, and source item identity. Our system focuses on maintaining independence from user interaction. In many systems (like Chapman's MiMI and others[4]) it is difficult to track information without maintaining complete control over how users interact with the information. To solve this issue, most systems take complete control of provenance objects and the data they associate with by not allowing users to interact with the data outside of the system's workflow. We sidestepped this issue by implementing a provenance library that is independent of the users' workflows. Though their list is desirable for most provenance systems, our project highlights an important feature of systems that make such restrictive provenance systems infeasible: independence.

B. CPL (Core Provenance Library)

CPL is a C-based library created by Peter Macko and Margo Seltzer of Harvard University. It currently supports bindings for perl, python and java with additional bindings for extra C++ features.[6] We chose to use CPL for our project as it allows ProvDMS to act independently of user workflows, unlike those of integrated provenance systems. By using CPL, we could define a system built on the trust of our researchers, allowing greater flexibility in the use of the data. CPL handles the logging aspects of provenance, automatically

keeping information about the objects we design in addition to the systems that own them and temporal information.

An important aspect of provenance is an efficient querying of information stored by provenance systems. CPL logs information via database (MySQL in ProvDMS's case). This allows our project to use a declarative approach for storing and retrieving information about our provenance objects.

IV. DESIGN

ProvDMS's design is both an object-oriented and a layered approach to interacting with the provenance library. We took these precautions to effectively separate the different aspects of our system so as to increase extensibility for later development. Figure 1 shows how these systems interact with each other. The raw data from our source (the FRPs) is held separate from the system, but it is logged to a database that we use as our initial back-end. This database is managed via LNDB (LoggerNet Database).[3]

A. System design

ProvDMS interacts with the data via a set of queries to LNDB. The information is directly utilized in the User Interface layer. Each action is used for tracking provenance. As the system needs to be non-restrictive, the users are not required to perform any special actions to allow for provenance. Instead, the system is built to detect actions and pass them to the compatibility layer (the wrappers).

Once in the compatibility layer, data transformations are involved to convert the user action into an appropriate call to the provenance back-end. The server handles all of the provenance requests asynchronously, allowing the client end to be lightweight in its processing. The project's requirements led us to design the system using PHP as the powerhouse behind transforming and transferring data between the provenance back-end and the user front-end. Javascript handles all of the information on the front-end and uses multiple libraries to provide a smooth and extensible system. The primary libraries used include: jQuery[5], W2UI[7], and Data Driven Documents (D3.js).[2]

The interface is designed using the class JavaScript Module Pattern, allowing for enhanced

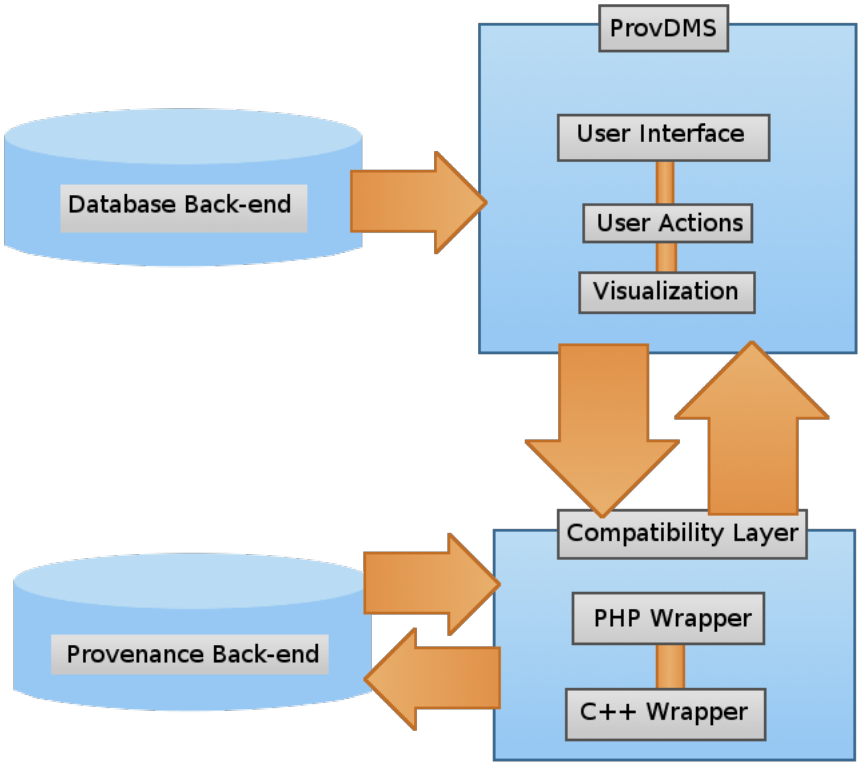

FIG. 1. Layer and compatibility design of ProvDMS. The system is designed to interact with a database back-end separate from the provenance back-end. Two layers of compatibility are built to interact with the provenance objects (PHP / C++ wrappers). The system also includes a visualization module built to interact with the compatibility layers.

extensibility. Each feature is designed independently of the others, but the modules are also designed to use a form of polymorphism, allowing for each module to be interchanged without fault. The C++ wrapper adds an extra object-oriented layer to the provenance API, giving a more defined and extensible layer for developers to use that is more understandable.

B. Provenance design

To keep the system independent from user workflows, we do not restrict information to any particular format. The provenance design for our system reflects the independent nature of ProvDMS in a few key ways:

1. Objects related to provenance are tracked from creation by both the data back-end and user front-end.

2. Modifications or translations to data are user-dependent. The system trusts the users to log information appropriately.

3. Provenance object derivation is handled entirely by the CPL back-end, with an emphasis on cycle avoidance and data flows.

The provenance object design follows a format that allows for both large and small granularity. A general sense of coarse granularity is maintained by the objects themselves while more specific information can be viewed via object properties (allowing for fine granularity in addition). These object properties give finer detail to provenance objects without being separate objects themselves. Figure 2 shows the design more in-depth for the FRP use case. When changing the system for other sources of data, this object inheritance hierarchy would change, but the same guidelines should be followed.

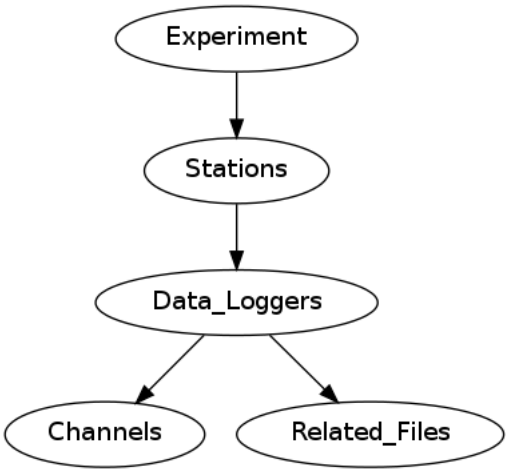

FIG. 2. Our system's provenance object design for the FRPs includes a parent Experiment object. This Experiment object can link to one or more Station objects via data flow. Each Station can have one or more Data Loggers associated with it and similarly each logger can have one or more "Channels" (sensors). In addition to the sensors, loggers also keep information about which CSV file was created as a result of the experiment definition.

An important aspect of our system is a low amount of overhead. Many provenance systems have difficulty limiting the amount of provenance information to store. MiMI[4], the provenance system discussed earlier, stored 270MB of data, but the provenance outstripped the data at 6GB before compression. The design of our objects and the session information

stored by CPL certainly takes space, but the overhead is less than traditional provenance systems. An in-depth analysis of this is needed.

V. FEATURES

ProvDMS includes a number of features to aid users in retrieving their information in a provenance-friendly environment. All aspects of the available user actions are unobtrusive and seamlessly integrated with the provenance back-end. We believe it is important to make the provenance process as user-friendly as possible without sacrificing any features of a provenance system.

A. Real-time retrieval

The ProvDMS system relies on a set of actions that asynchronously work on real-time data to build both provenance information and usable data.

1. Experiment creation

The first interaction involved with the data is between LNDB and the ProvDMS interface. As all requests are asynchronous, none of the data is directly accessed until a user decides they want to begin a user action. For interacting with the original data, this involves the definition / creation of an experiment via the user interface. The ProvDMS experiment creation module allows users to select particular stations, data loggers and sensors from the data source (queried from LNDB). Each sensor brings a host of metadata available to the user as well. This metadata is logged and used to specify the granularity of the provenance via properties during visualization.

Users are given the option to define particular time-ranges for their data upon creation. This information is saved as provenance properties and can be used by the back-end to index provenance information by experiment time-range. The declarative nature of the database allows this feature to be quite efficient.

2. Experiment import

In addition to new experiments, users are also allowed to import derived experiments. Provenance requires experiments that are imported be associated with a previous experiment. This particular feature of our system places trust in the users to ensure that proper provenance information is identified and stored by the system. Each file that is associated with the imported experiment allows the user to add an additional property via comment.

B. Security and XCAMS

Many hardening steps have been taken to ensure the system is secure. The following features were carefully detailed in the design and development of ProvDMS:

1. Prevention for session hijacking by assuring proper secure sessions are used and stored as encrypted cookies.

2. Protection against SQL injection attacks via the use of prepared statements and input sanitation.

3. File access protection via secure POST and concurrent file protection using a temporarily available database link to exported files. Schedule jobs handle the cleanup of temporary entries and files to ensure both the server and database remain efficient in their indexing and storage.

4. Secured connection via HTTPS (SSL) for encrypted site traffic.

In addition to the precautions taken above, we have designed the system to use XCAM authentication as part of the user system. User login is dependent on a particular token being returned from ORNL's XCAMS system, ensuring only valid users may access our ProvDMS system.

C. Visualization

A very import feature of our provenance system is allowing users to see an overview of their experiments' provenance. To achieve this, we use D3.js[2] in combination with our compatibility layer (the wrappers) to retrieve ancestral information from particular provenance

objects. This information can be parsed and used to display varying types of graphs for visualization of the data.

A key aspect of our visualization involves a cleaning process for retrieved information to skip certain unnecessary ancestral links for a simplified user experience. These links are necessary for laying out provenance data, but obfuscate typical visualization techniques. As a result, we limit our visualizations to show only one version of each provenance object. Therefore, objects of type Experiment will be linked to stations and newer versions of Experiment objects, while stations will only be linked to Data Logger objects. It is only necessary to visualize the display information directly related to the experiments due to the granular view of the provenance information. If this cleaning process were not used, information would be very generally shown. A user could see data flows from the beginning of a particular experiment to a station or logger in an entirely different experiment. This is not a flaw, it is a result of CPL's versioning system — the reason our system enjoys a low amount of object overhead.

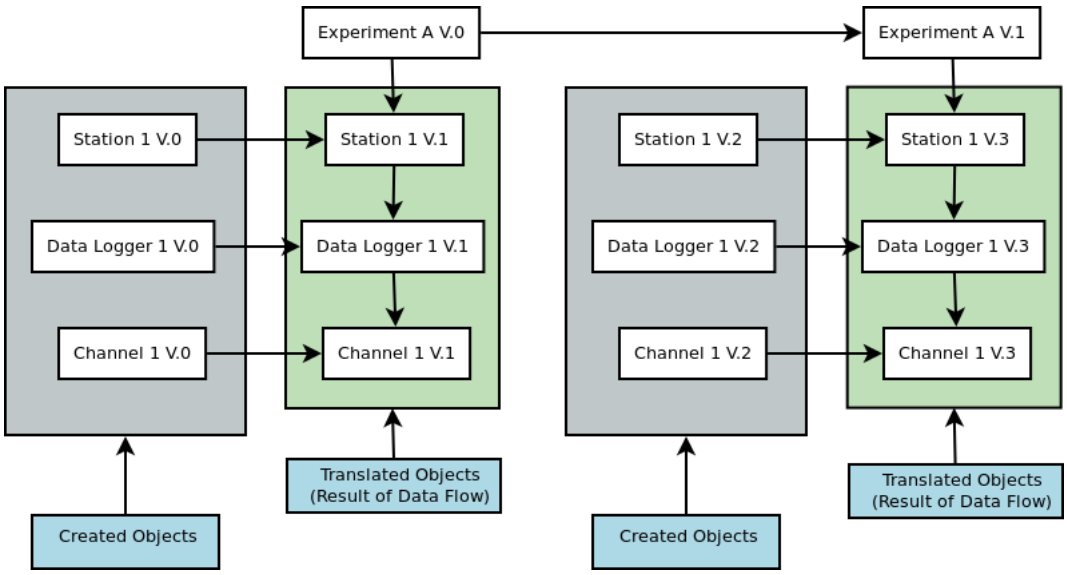

FIG. 3. Logical representation of a subset of provenance data. Two versions of the finer granularity objects exist as a result of data flow dependencies and the Cycle Avoidance algorithm. These extra nodes and others must be parsed out for clean visualization of the provenance.

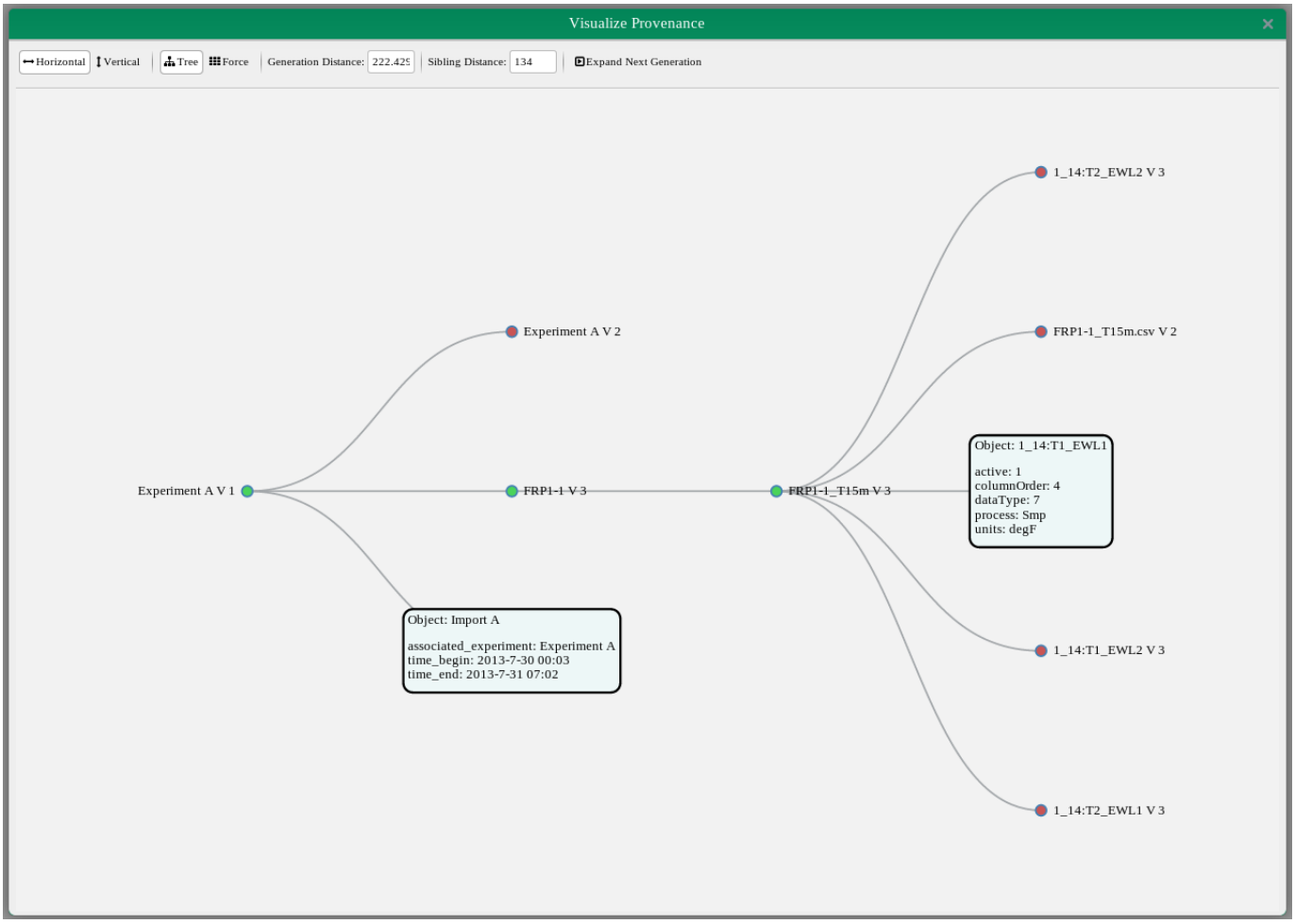

FIG. 4. Node-Link, Contextual Tree layout visualization of provenance data. Figure shows a few expanded nodes with fine granularity combined with contextual information. This is the current visualization used with ProvDMS.

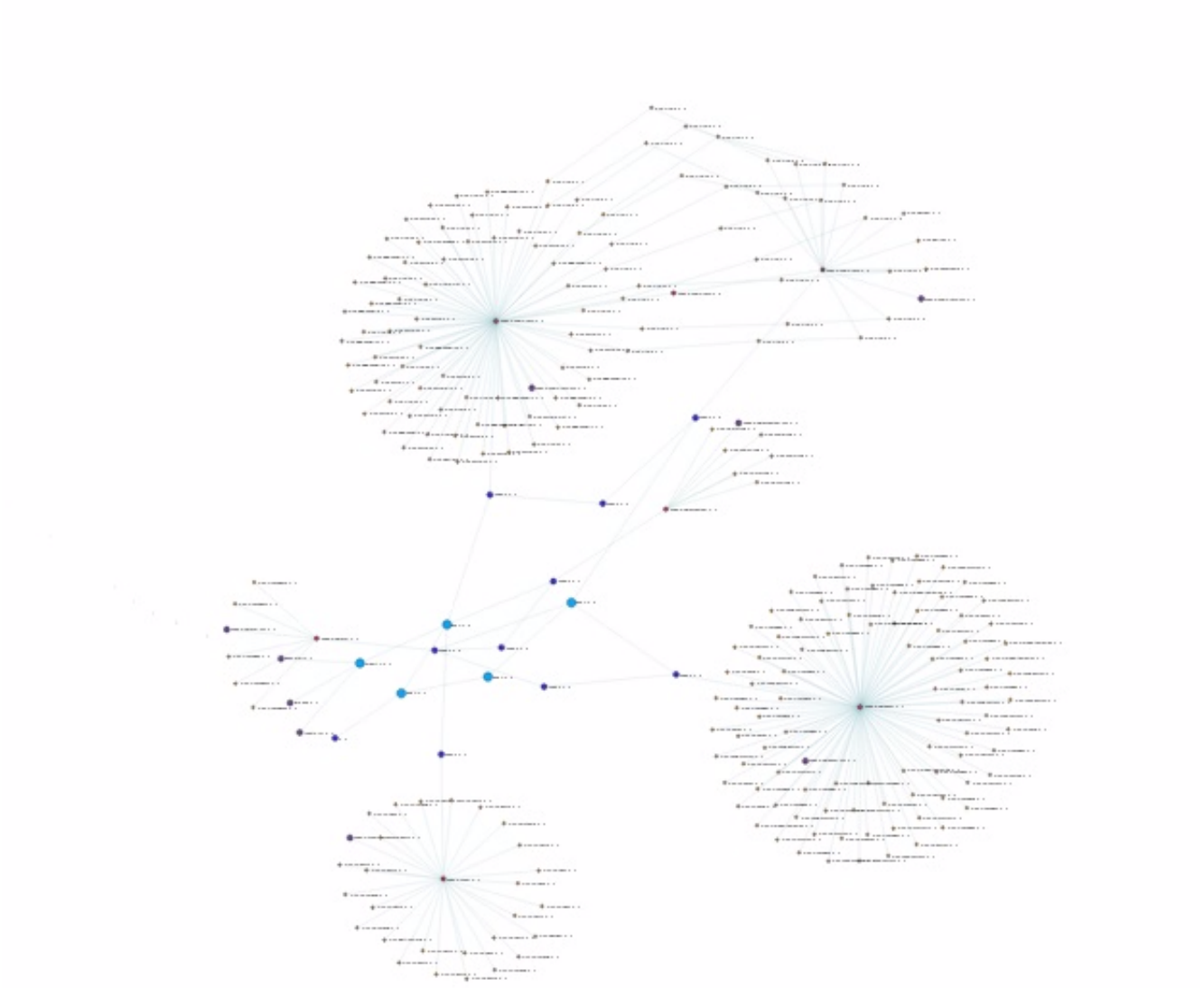

FIG. 5. Node-Link Force-Based layout visualization of provenance data. Figure shows overview of provenance data for a particular experiment and its lineage.

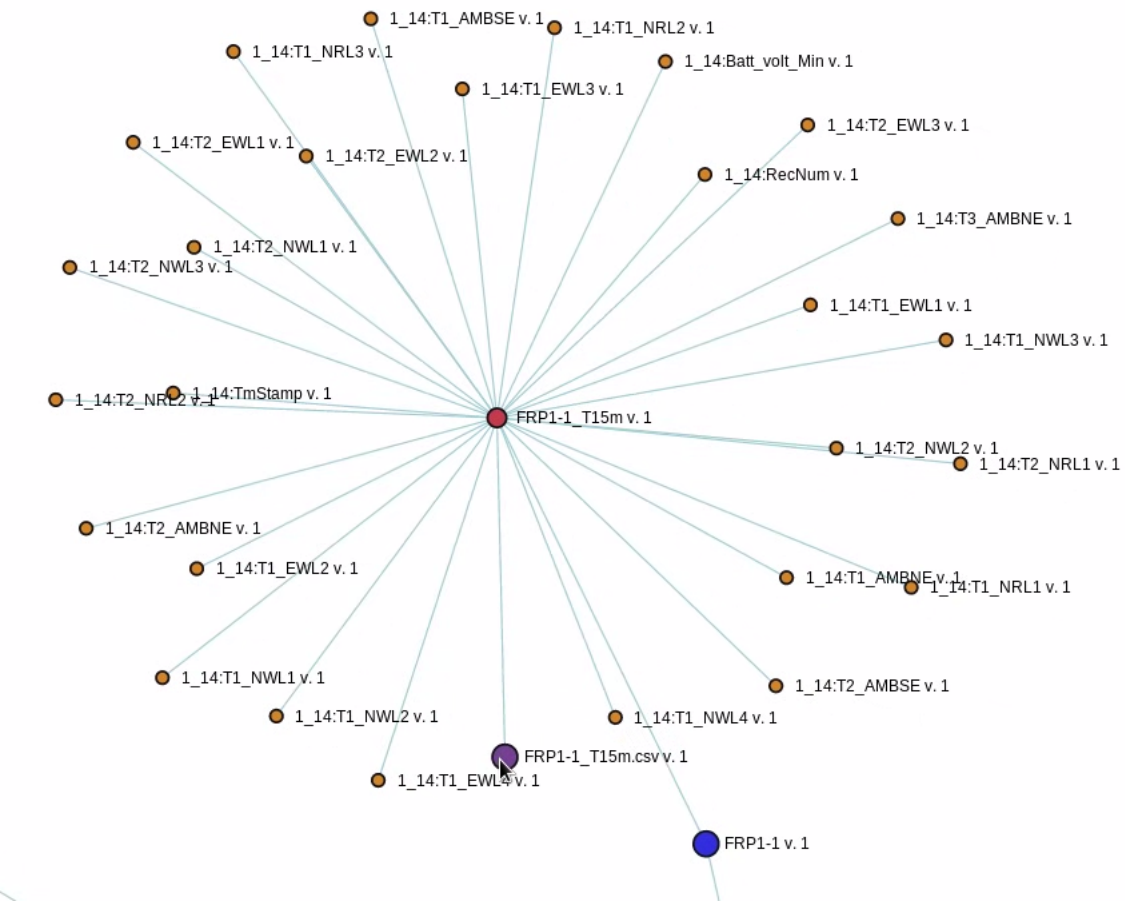

FIG. 6. Close-up of a grouping of fine granularity objects in the node-link force-based layout visualization.

VI. DEVELOPMENT PROCESS

A. Agile development

We employed an Agile Development process according to Scrum when designing and developing our system. Our Sprint time was set at two weeks. This meant we held biweekly meetings to determine the important issues to use as our Sprint backlog (a subset of the project backlog). We met daily (unless impeding circumstances) to discuss important project information and focus points of the Sprint backlog. To help in managing this particular development process, we used Pivotal Tracker to assign a set of stories to focus on for each Sprint session.

B. Challenges and collaboration

Each aspect of the development process had a list of challenges. Many different libraries were tested for their potential inclusion in the user interface. The objective of deciding which to use as the primary backbone of the interface was the first step.

The provenance back-end hosted a number of challenges for use in our system. CPL is designed to pull particular information from the executing environment to log session data for object provenance. This meant it was important for our system to properly detect information for CPL to use. As our system runs from a server environment (passing through wrappers), it doesn't have all of the required information to properly log provenance session information. We patched this issue by directly passing important environment information to the provenance backend.

An unexpected challenge involved differences between test data and production data. The project was initially designed using a test data source and a local environment for the hosting of the system. Once we were ready to migrate to the deployment stage, it was necessary to restructure parts of the system to properly adjust. This involved many query rewrites and schema changes to properly represent the new data.

By far, the most challenging aspect of developing ProvDMS was the integration of CPL's Cycle Avoidance system. This pertains to the method with which CPL handles versioning of provenance objects so as to assure each object is unique in its indexing. CPL keeps a low overhead of time and space by keeping the number of indexed objects low in comparison to how many different provenance objects are logically present. This idea is perfect for data translations or transformations. By using Cycle Avoidance, new versions of objects are created when data transformations occur, shown using links known as *data flows*.

If we were to create a provenance object "Table 1" and wanted to modify this table, it would be important to differentiate between the original object and a new version of this object when we want to further modify this table. This means that, when we modify "Table 1", we create "Table 1 version 2" that links back to "Table 1".

This whole process is nice for the logical design of our objects; however, our process of creating and linking objects and the Cycle Avoidance algorithm creates new challenges. When our system creates new objects, we need to ensure these objects link to the proper versions of parent objects. New *Station* objects should properly link to a particular version of

Experiment and so on. What this means outside of the logical view of our objects, however, is a poor integration with the Cycle Avoidance algorithm. Instead of a singular object being created for the *Station* object, two must be created. The first is a new instance of the latest version of the *Station* object. The second is another *Station* object that handles the data flow link of the *Station* object back to the *Experiment* object. As a result of this Cycle Avoidance behavior, we must take precautions to properly parse retrieved information from the provenance back-end.

VII. CONCLUSION

Provenance systems are used in many forms of research, each with some particular disadvantage. Our ProvDMS system eliminates issues relating to tool restriction and allows users to be more flexible in their approach to research data. With the use of CPL as a provenance back-end, we have been able to separate provenance responsibilities from the data access system and ensure a low overhead is used — making the system more efficient.

As a result, our provenance system can handle many forms of granularity without noticeable slowdown. Our wrapper compatibility layer allows us to handle provenance information in a general format which can be used for accessible visualization of the provenance information for our users. The manner in which we handled ProvDMS's object design allows for contextual information to be given for desired objects.

The results of our work on ProvDMS allows approved researchers to better track how their data is used and how it changes over time. The specific work on independence allows us to further extend and modify the system for new tracking methods and integration with user tools without adding new restrictions on user interaction. Our approach of separating the provenance back-end from the user front-end and data back-end has proven to be a useful and effective measure of providing non-restrictive provenance to our researchers. We see ProvDMS and the cohesive, but independent design of ProvDMS as applicable in many research applications.

ACKNOWLEDGEMENTS

The Science Undergraduate Laboratory Internships (SULI) program is sponsored and managed by the DOE Office of Science's, Office of Workforce Development for Teachers and Scientists (WDTS) in collaboration with the DOE National Laboratories.

The research work was performed at the Oak Ridge National Laboratory, which is managed by UT-Batelle.

The authors extend their thanks for the generous work of Dr. Margo Seltzer and PhD candidate Peter Macko of Harvard University for their work on CPL and collaboration during this project's term.

The authors would like to thank the Energy and Transportation Sciences Division for the wonderful chance to work together on this project. Further thanks go to Julie Malicoat and Cheryl Brown for their work with the 2013 interns and the SULI program.

REFERENCES

[1]Mahabir S Bhandari and Heather L Buckberry. Ornl maxlab flexible research platforms. Technical report, Oak Ridge National Laboratory (ORNL); Building Technologies Research and Integration Center, 2012.

[2]Michael Bostock. Data driven documents. web, August 2013.

[3]Campbell Scientific, Inc, 815 W 1800 N — Logan, Utah 84321-1784. *LNDB LoggerNet Database Software*, September 2010.

[4]Adriane Chapman and HV Jagadish. Issues in building practical provenance systems. *IEEE Data Eng. Bull.*, 30(4):38–43, 2007.

[5]The jQuery Foundation. jquery - javascript library. web, August 2013.

[6]Peter Macko and Margo Seltzer. A general-purpose provenance library. In *Proceedings of the 4th USENIX conference on Theory and Practice of Provenance*, TaPP'12, pages 6–6, Berkeley, CA, USA, 2012. USENIX Association.

[7]Vitali Malinouski. W2ui - javascript ui library. web, August 2013.