Adapting the Electronic Laboratory Notebook for the Semantic Era

Tara Talbott[1], Michael Peterson[1], Jens Schwidder[2], James D. Myers[1]
[1]*Pacific Northwest National Laboratory*
[2]*Oak Ridge National Laboratory*
Tara.Talbott@pnl.gov, *Michael.Peterson@pnl.gov*, *schwidderj@ornl.gov*, *Jim.Myers@pnl.gov*

ABSTRACT

The open source Electronic Laboratory Notebook (ELN) is a collaborative, distributed, web-based notebook system, designed to provide researchers with a means to record and share their primary research notes and data. As with most electronic notebook (EN) systems, the ELN was originally designed as a closed system with its own data repository and implicit semantics. The Scientific Annotation Middleware (SAM) project, a Department of Energy (DOE)-funded effort at Pacific Northwest and Oak Ridge National Laboratories, envisions a new model in which ENs are simply one application contributing to a much richer and semantically explicit record. Such a record would include, for example, data provenance, descriptive metadata, and annotations from a wide range of applications, problem solving environments, and agents. This paper reports the adaptation of the ELN client to use SAM and the development of an initial set of SAM-based notebook services and semantic model, and then discusses the advantages of such an architecture in creating federated, human- and machine-interpretable, electronic research records.

KEYWORDS
Electronic Laboratory Notebooks, Scientific Annotation Middleware, Collaboration, Semantic Data Management

INTRODUCTION

Traditionally, scientists have used paper-based laboratory notebooks to record their ideas, observations, and data. Regulatory and scientific standards require notebooks for purposes of intellectual property protection, but they also have value as a knowledge base of previously performed work. In recent years, many organizations in research-intensive fields have begun to look closely at Electronic Notebooks (ENs) as a new way to record, store, and manipulate experimental data. ENs have some compelling advantages over paper-based notebooks. They allow scientists to work in distributed teams, sharing data with peers in real time, and receiving comments on their research. ENs eliminate the need for manual transcription of data that already exists in electronic form and they can directly display the large, multidimensional, and time-dependent data sets produced in modern experiments. Further, they can have automated searching, indexing, and metadata generation capabilities to aid in enterprise knowledge discovery. ENs also have advantages in terms of legal defensibility because of the strength of digital signatures and as records because of the low cost of digital media storage. Thus, ENs become much more than just a record-book—they provide a collaborative, intelligent working environment.[1]

Although an EN has many advantages over paper notebooks, the model of an EN as a stand-alone system has become limiting. While stand-alone ENs can associate manually entered notes with output and analysis from scientific instruments on a single page view, they are limited in their ability to interact with other producers, curators, and consumers of data and annotations. Feedback from users of the Electronic Laboratory Notebook (ELN), as well as developers of other types of scientific software, indicates growing interest in ENs capable of acting as a component in more comprehensive semantic systems incorporating components such as instrument control software, problem solving environments (PSEs), autonomous feature-detection agents, digital libraries, and data pedigree mechanisms.[2] While a number of notebooks, ranging from the Spectro-Microscopy EN developed in the mid-1990's [3] to those based on the CENSML

language [4] and to commercial notebooks such as the Rescentris CERF/Notebook [5], incorporate a notebook schema and can be extended with science-domain-specific schema, the concept of ENs interacting with other components as equals annotating shared resources has not been fully explored.

The Scientific Annotation Middleware (SAM) is middleware designed to manage semantic information directly and provide a shared 'schema-less' metadata store usable by multiple annotation and records-related applications to create rich scientific documentation. We report here on work to modify the ELN Client to use SAM as a notebook server and discuss the initial benefits of this model of using an EN with a general semantic store. In updating the ELN, we had to consider how to represent the ELN data model with explicit semantics and define a mapping between the ELN client-service functionality and SAM's existing API, aiming to simplify and standardize communication between client and server. The result is a new generation of the ELN capable of storing data in a variety of underlying repositories, exposing its metadata in the resource description framework (RDF) syntax, and integrating much more deeply with other systems producing and consuming data and metadata.

BACKGROUND

ELN Functionality:

The overall functionality of the ELN, and its communications architecture, has previously been reported in depth [6]. In summary the ELN provides a chapter/page-based interface to an EN in which users can login and interact with the data. The functional core of the ELN circa version 4.6 consists of an interactive browser-based client and a custom web- based notebook server written as a set of Perl CGI scripts.

Users are allowed to view, add, or edit data in the notebook as well as perform tasks such as search and user management. Data can be added in multiple formats such as text, equations, images, or uploaded files. The ELN supports plug-in mechanisms to support adding and 'viewing' new data types, e.g. capturing voice annotations and displaying a playback interface within the notebook page. Once data is submitted, it is immediately available to other authorized users for viewing and annotation. The ELN structure is similar to that of a paper notebook, consisting of a hierarchy of chapters, pages, and multiple levels of notes containing data objects and sub-notes. Each notebook object includes either a blob of binary data or a list of child objects along with an extensible set of metadata. The minimal set of metadata includes items such as author, creation date, content type, and size. In the Perl-based ELN server, the data is transmitted between the client and server using HTTP GET and POST methods using ELN-specific formats. On this server, metadata is stored in files using a custom format, hierarchy is encoded in the server's local directory structure, and data is stored as files with the same content as the submitted blobs.

Scientific Annotation Middleware (SAM):

SAM, being developed by researchers at the Pacific Northwest and Oak Ridge National Laboratories, is a layered set of components and services for managing data annotations and the semantic relationships between data objects. [7] SAM is a schema-less store that can manage arbitrary metadata and relationships that are defined by namespace qualified names. As such, it is well suited for a written-by-one-read-by-many usage model in which multiple applications contribute unique information about different aspects of data, all of which must be presented to the user and further analysis tools as an integrated data context.

SAM is built on the Jakarta Slide content management system and its web Distributed Authoring and Versioning (webDAV) protocol implementation. WebDAV and its extensions adopt the Web's HTTP model of resources accessed via a URL, adding standard methods for creating new collections (directories) and resources, adding and querying name–value-pair properties (arbitrary strings or XML) associated with each resource, and supporting versioning, locks, and list-based access control. [8,9] Slide implements a webDAV-centric content repository as middleware that can store metadata and data in multiple independent underlying data stores. When new resources are created using webDAV, Slide generates standard webDAV properties that describe the resource, such as its type, size, owner, and creation date. As part of SAM development, we have contributed code and discussion to the open source Slide effort, reporting bugs and submitting fixes and performance enhancements.

SAM also extends Slide in a number of ways that enhance its ability to function as an integration mechanism. To make activities in SAM visible to third-party software, we have modified Slide to produce Java Messaging Service (JMS) events whenever a resource is accessed or modified via webDAV. SAM publishes

events under two JMS topics, one for changes to the data or metadata (e.g. PUT, PROPPATCH, and COPY requests) and one for reads (e.g. GET and PROPFIND requests). (Events contain an adjustable level of detail about the request to aid in filtering messages, but a subscribing application must access SAM directly, and have appropriate permissions to access the specified resource and retrieve the viewed or modified content.)

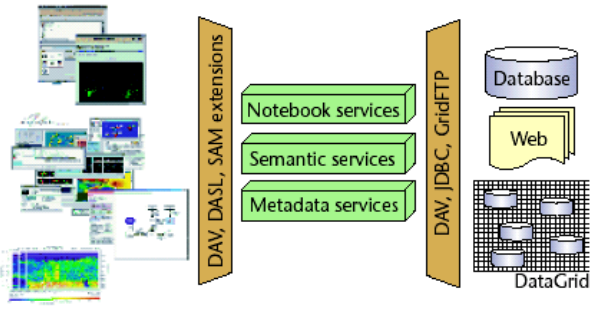

Figure 1. Scientific Annotation Middleware (SAM) service layers presenting a federated view of data to applications, portals, agents, and electronic notebooks.

SAM also provides configurable, automated metadata extraction and translation of uploaded resources. Combinations of XSLT and Binary Format Description (BFD) scripts and web services can be registered with SAM and run dynamically to extract metadata or create translations and data views from binary, ASCII, and XML inputs. BFD is a language used to describe the structure of a binary or ASCII file and how it should be parsed to produce the desired output in XML. For example, when a binary data file is uploaded to SAM, e.g. by dragging the file into a networked disk drive which uses webDAV, SAM can use a registered BFD file describing the format to translate the file into XML and then use a registered XSLT script to generate relevant webDAV properties. Similarly, SAM can use these same mechanisms to generate translations and views of data - for example it can produce static HTML pages or pages invoking Java applets for a browser-based view of the data. SAM reveals the translations available for a given resource using a "hastranslations" property specifying 'virtual' URLs for the translated content. Translations in SAM are created dynamically, instantiating the translation URLs when they are requested rather than generating and storing all possible translations in advance.

Supplementing Slide's default internal authentication method, we've added a Java Authentication and Authorization Services (JAAS) based mechanism to allow SAM to be configured to use an external authentication service. As part of this mechanism, SAM supports one-time tokens and credential caching allowing, for example, one application to pass a token to another to allow it to access data in SAM without revealing the user's long-term username/password, certificate/private key, etc.

IMPLEMENTATION OF ELN-on-SAM

By definition, a service capable of managing arbitrary metadata and relationships is sufficient to manage information associated with any specific data model. Hence, in theory, SAM's metadata management layer and the webDAV protocol could be used directly as a notebook server. In practice, it is convenient to define a higher level interface for operations that would otherwise involve multiple rounds of client-server communication. In the case of the ELN, which relies on notebook page displays that are rendered in the browser, it is also convenient to add page rendering capabilities to the server. The original design concept for SAM included additional layers of general capabilities related to managing information in semantic terms and specifically within annotation and records-centric applications. Migrating the ELN to work on SAM, as detailed below, represents the initial work to define and implement these layers. Thus, our goal included not only duplicating or expanding the capabilities of the ELN 4.6 server, but also considering how to abstract the ELN data model and capabilities.

Before tackling the main task of creating an updated server, we re-implemented portions of the ELN client to improve encapsulation of client-server communications, adding a reflection-based mechanism to allow the client to dynamically load classes for communicating with different server types. Although the ELN had a model-view-controller architecture, additional work was required to organize code for communicating with the local browser (e.g. for requesting display of a specific notebook page) and the server (e.g. for adding a new note), to factor out reusable code, enable reflective discovery of available communications classes, and create a central mechanism for dynamically selecting appropriate communications classes for a given server. The reflection mechanism was used initially as a means of continuing support for the 4.6 server while developing a SAM and webDAV based communications library, but represents a general mechanism that can be used to adapt the ELN client for use with additional servers.

The results of this redesign are the areas of functionality and methods shown in Figure 2. This represents the full scope of the ELN's client-server interactions in terms of abstract operations. To realize these using SAM and the webDAV protocol required two steps: mapping the ELN data model to the webDAV resource-plus-properties model and mapping the specific methods to atomic server calls.

Server Information	getNotebookUID() getServerURL() getEditorClassNames()
Notebook Retrieval/Submission	login() logout() getNObs() addAnnotation() updateNOb() delete()
Notebook Configuration/Information	addNewUser() removeUser() resetUserCredentials() getUserList() getUserRole() getBackgroundColor() isNotificationEnabled() getInterestList() setInterestList()
Page Display	getPageDisplayer() displayAboutPage() displayHelpPage() displayHomePage() displayNotebookPage() displaySearchResults() closePage() closeAll()
Sam Specific Functions	getRootNode() requestPage()

Figure 2: Classes and methods representing the full scope of the ELN's abstract client-server interactions.

To a large extent, the ELN data model maps directly to the webDAV model (in fact, experience in developing the ELN led to the choice of webDAV for SAM). The primary challenge arose in making aspects of the data that are implicit in the ELN explicit in SAM. For example, while the ELN client represents property names as strings and assumes their meanings, we mapped them to fully qualified namespace/name pairs in SAM. In most cases, we mapped ELN property names to existing, standard qualified names, e.g. taking the "DAV: creationtime" to represent "creationtime" and from the Dublin Core Initiative, an open group for the development of metadata standards [10], http://purl.org/dc/elements/1.1/creator (dc:creator) to represent "author". For any information where the correspondence with an existing term was not clear, e.g. to support properties created by third-party ELN editors, we created a SAM notebook namespace "http://purl.oclc.org/NET/SAM/ns"(samns).

One important place where this namespace is used is in specifying the hierarchical notebook relationships. While we could have modeled the notebook hierarchy solely using webDAV collections, we did not do so for two reasons. First, webDAV collections, like file system directories, do not have any explicit meaning beyond 'contains' and their use is highly overloaded; people may group files based on similar content, chronology, or association within projects, while applications may also use directories to structure their output based on their internal data models. While the organization of material within the notebook is still somewhat a matter of personal style, we felt that, as a minimum, we needed to distinguish the hierarchy defined within the notebook application from that of other applications. Additionally, we also anticipated use of the ELN with other applications sharing the SAM data space and with the ELN referencing information uploaded by those applications (i.e. using the ELN's URL editor). Since the ELN and other applications would not be able to both have independent control of the webDAV collection hierarchy, we decided to explicitly represent the ELN hierarchy via a property (samns:children). We also decided to explicitly represent which notebook the hierarchy belonged to, leaving the door open for future functionality where multiple notebooks might have overlapping content, e.g. a private notebook containing the contents of a public notebook(s) along with some additional notes.

Using a single, multi-valued property, rather than individual "samns:child" properties is dictated by webDAV, which requires that properties have unique names. We chose to represent the set of child relationships as an XML value of the samns:children property consisting of samns:notebookroot elements that identify specific notebooks enclosing individual samns:child elements identifying the individual relationships. (Although it is not necessary for the operation of the ELN, SAM is capable of representing this property as RDF, using reification to identify which samns:child relationships belong to each notebook.)

As written, the ELN client still distributes notebook content into a webDAV collection hierarchy, but this choice was arbitrary (performance of early versions of Slide degraded as the number of resources in a collection grew, but this issue has been resolved) and, given the samns:children property, we could have chosen to store everything in one webDAV collection. Further, if we change this choice, or if we implement the functionality to include a chapter from one notebook in another (stored in different webDAV collections), content for a given notebook may be distributed across multiple webDAV collections yet notebook navigation will be unaffected.

These conventions are sufficient to represent the core ELN data model in SAM. In adapting the ELN to SAM, we also chose to represent notebook configuration information, which had previously been stored in custom-format files on the server, as a webDAV resource with properties, e.g. with a samns:bgcolor property to specify the color of notebook pages and a samns:notebookroot property pointing to resource representing the top of the hierarchy. Thus, with SAM, the full state of a notebook is explicitly represented and available to any webDAV-capable application/user with sufficient access permissions.

Given these choices for representing the ELN data model, most of the client-server functions could be reduced to a single or short sequence of webDAV calls. For example, to create a resource you would use the webDAV PUT method and to set its properties you would follow with the PROPPATCH method. In simple cases like these, e.g. for creating and retrieving notebook content, we have implemented the required calls in the client. However, three types of methods required additional server-side functionality. We developed Java Server Page (JSP) based servlets for methods involving server-generated pages, e.g. for displaying notebook creation forms, showing a list of notebooks available to a user, or generating notebook pages. For methods querying or setting notebook configuration information, we implemented minimal server-side wrappers that were equivalent to webDAV calls but allowed access to be controlled on a per resource and per-property basis. While these methods could have been left as webDAV calls from the client, we felt that per-resource access control, which would be possible with SAM, could potentially expose more information than would be desirable and that implementing general per-property access control in SAM would require significant work. Hence, since we felt that users in general would not require access to the configuration information directly via webDAV, i.e. not through the ELN, we implemented servlets that provide a lightweight wrapper around the property get and set operations that can limit access to users of a given notebook and to properties required for operation of the client. (Direct webDAV access can still be granted to the configuration resources if a need for this is identified, but, by default, we deny access.) Lastly, we implemented a new class of methods for configuring notebooks via the browser that duplicate and extend the configuration capabilities available through the ELN client. (This was done partly in anticipation of a fully web-based ELN implementation which is currently under development.)

In several areas, SAM's general capabilities greatly reduced the notebook specific programming required as highlighted in Figure 3. For example, the ELN's pluggable viewer mechanism, used to define how various types of annotations and data files are displayed in the notebook page, is simply a special case of SAM's more general translation mechanism. Thus, in writing a page display servlet for the SAM-based ELN, we had only to write logic to display the overall page structure and then invoke the existing functionality to render individual entries. (Since SAM allows multiple translators per data type, the choice of which one to use in the ELN is needed as part of a notebook's configuration.)

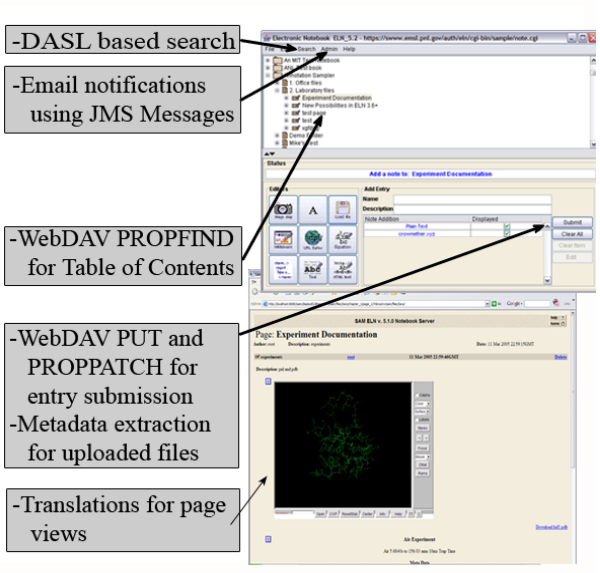

Figure 3: View of ELN Client and Page View showing functionality which used to be ELN-specific but is now generic in SAM.

Similarly, the Perl-based ELN server had its own email notification system that required an external 'cron' capability to periodically launch the email mechanism and a mail daemon to actually send the messages. With SAM, rather than parsing a log file to discover notebook changes, it is possible to monitor JMS events. Further, we were able to reuse the Notification Email Daemon (Ned) code developed for the Collaboratory for Multiscale Chemical Science (CMCS), which was designed to monitor data updates created by a variety of tools, to filter and digest events and send email notices (via the Java Mail API) [11] A researcher using the notebook can subscribe to receive notifications of all changes occurring in a particular chapter or on a particular page and the email service then filters for JMS events in which the samns:children property is updated and, depending on subscriber preferences, sends an email notice out immediately or digests it with other changes to be reported at regular intervals (configurable). Conversely, the use of JMS provides a standard mechanism for other applications to monitor notebook activity and for the ELN to be incorporated into larger scientific workflows.

Search is another area where the ELN is able to leverage underlying capabilities in SAM. SAM supports the DAV Searching and Locating (DASL) standard which defines a SEARCH method and basic SQL-like grammar for queries against webDAV repositories. SAM has developed an extended grammar that allows searches to be scoped based on the pattern of relationships between resources, i.e. to search through all resources linked to the current one by a specific property. For the ELN, we were able to replace the Perl-based search routines with a light wrapper around the SAM DASL implementation and specify that searches should be scoped by samns:children links to support searching within a notebook, chapter or, page. In addition to simplifying the ELN search implementation, the use of DASL and explicit semantics makes the notebook content and structure available to other applications enabling, for example, a PSE to be configured to infer a project relationship between data sets in the same chapter without a need for prior agreement between the ELN and PSE developers

DISCUSSION

Re-engineering the ELN to work on top of SAM has been very successful, resulting in a system that is more functional, more maintainable, and significantly more capable of integration into larger systems than earlier versions. It has also demonstrated the advantages of, and some issues with, a semantic middleware model as realized in SAM.

As an EN, the ELN-on-SAM is much more powerful and flexible than previous versions. It can be run using a file system as a data store, or migrated to open source or high-end commercial databases for improved performance and scalability, or potentially integrated with a full records management system. Data within the notebook is now readable using webDAV browsers such as DAVExplorer [12], as files on a shared drive (e.g. using webDrive [13]), and directly through the web. Further, the ELN can directly reference data and metadata created by other tools, i.e. pointing to data saved by instrument control software on a SAM-based shared disk drive and sharing the dc:creator metadata specifying the author.

The changes have also made it easy to integrate the ELN into portals such as those developed for the George E. Brown Network for Earthquake Engineering and Simulation Grid (NEESgrid) and CMCS. In CMCS, which uses SAM as its primary data repository, the integration is quite deep - for example, notebook annotations appear as part of the data provenance graph the portal can generate (Figure 4), and the ELN can annotate data generated by any CMCS portlet, application, or web service.

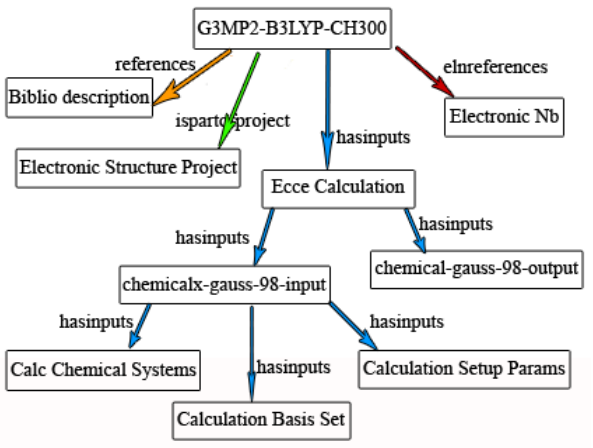

Figure 4: View of pedigree graph in CMCS portal showing notebook references along with relationships written by other applications, generated from dynamically generated property in SAM. (Additional relationships detailing the internal notebook structure are not shown in this figure.)

In terms of maintainability, by shifting functionality to middleware, the ELN-specific server-side code has been reduced by roughly 50% and the ELN benefits from enhancements developed within the Jakarta Slide project and from efforts related to the development and use of SAM. For example, the SAM-based ELN has already received a significant performance boost from work within the Slide project to implement caching and to incorporate a Lucene-based index [14] into the search method.

In general, the changes we made to the ELN to explicitly define its data model in semantic terms and to expose that information to other applications via webDAV were straight forward and added minimal overhead. However, two issues deserve further discussion. The first relates to the lack of support for multi-valued properties in webDAV (or for a similar capability such as support for multiple properties with the same qualified name). While standard conventions for representing multi-valued properties in SAM allow them to be interpreted correctly by collaborating applications, the lack of methods to add, modify, or delete specific values within a property makes it necessary to include the logic to manage multi-valued properties in the ELN client and results in an extra call to the server to retrieve the current set of values before adding a new one. We are currently investigating ways of addressing this – by creating a non-standard extension to the webDAV PROPPATCH method or providing an additional interface to SAM with methods dealing directly with individual semantic relationships.
The second issue is more philosophical. By using webDAV rather than a custom ELN server interface, we've lowered the barrier to other applications writing ELN information, i.e. adding samns:children properties. While this can be desirable, and has been used, for example, to create a non-notebook annotation tool [15], it also raises the concern that other writers might not fully share the semantics the ELN client associates with the properties it writes. In some sense, this concern exists in any client-server application – third-party clients could use the server interface in unintended ways – the use of a general middleware interface raises its prominence. As noted, we have developed some non-webDAV methods to reduce this concern for ELN configuration information, but we are also considering the option for a more general mechanism, i.e. enhancing SAM to limit which clients (as reported via the Http-client header) can write information in specific namespaces such as samns, or allowing SAM to dynamically infer equivalence between different relationships (supporting a model where different clients would write to their own namespaces and rely on SAM to map relationships identified as equivalent into their own namespace for reading).

CONCLUSIONS

We believe that exposing the data and activity of ENs through standard metadata representations and protocols will be an important step in bringing them into the broader world of the semantic web and knowledge grids. With growing automation of metadata capture through instrument control software, laboratory information management systems (LIMS), PSEs, workflow tools, and other mechanisms, and the growth of projects spanning institutions and disciplines, ENs will no longer be the sole source of, or interface for, scientific records to the extent that paper notebooks have been. However, we believe ENs capable of contributing to a shared record with explicit semantics will be a key enabler of next-generation research

The effort to migrate the ELN to use SAM provides a concrete example of the work involved to begin such a transformation. Our experience to date in using the SAM-based ELN as a standalone EN, along with the experiences of others in integrating it into portal systems, demonstrates some of the advantages of this design as well as some directions for future work. Most broadly, we intend to explore the ability to have the ELN interact with other applications by, for example, configuring the ELN to infer chapter/page relationships from project/experiment relationships recorded within a PSE. We also plan to develop a next-generation user interface for the ELN that will provide an updated look-and-feel and make it easier to incorporate metadata created by other applications into the ELN page display. Lastly, we are working to migrate the ORNL eNote notebook and Platypus Wiki to use SAM. We are hopeful that all of these can be supported with a common set of server interfaces and shared semantic data model that can, in turn, inform efforts to develop a general annotation architecture for semantic grids.

ACKNOWLEDGEMENTS

This work was supported as part of the Scientific Annotation Middleware (SAM) project. Employees of Battelle Memorial Institute, which operates Pacific Northwest National Laboratory for the US Department of Energy under Contract DE-AC06-76RL0 1830 and Oak Ridge National Laboratory under Contract De-AC05-00OR22725 wrote this manuscript. The authors would also like to acknowledge the efforts of Prasad

Saripalli, who contributed to an early draft of this paper, and other past and present members of the SAM project team. The authors also acknowledge helpful discussions and ongoing collaborations with members of the Collaboratory for Multiscale Chemical Science (CMCS) project who have helped define requirements for SAM.

REFERENCES

[1] Lysakowski, Rich, "Comparing Paper and Electronic Laboratory Notebooks," http://www.censa.org/html/Publications/Comparing_Paper_and_Electronic_Lab_NotebooksP1_SC&A_May-1997.html

[2] J. Myers, E. Mendoza, A. Geist, "Scientific Annotation Middleware Technical Overview," June 2001. http://collaboratory.emsl.pnl.gov/sam/samtechoverview.html

[3] Sonia R. Sachs, Carla M. Dal Sasso Freitas, Victor Markowitz, Anna Talis, I-Min A. Chen, Ernest Szeto, Harumi A. Kuno, "The Spectro-Microscopy Electronic Notebook" LBNL Technical Report LBNL-39886, http://http.icsi.berkeley.edu/ftp/global/pub/techreports/1997/tr-97-002.pdf

[4] Collaborative Electronic Notebook Systems Markup Language (CENSML), http://www.censml.org/index.html

[5]Rescentris CERF™ and CERF/Notebook™ Overview, http://www.rescentris.com/pages/1cerf.html

[6] J. Myers, E. Mendoza, and B. Hoopes, "A Collaborative Electronic Notebook," *Proc. IASTED Int'l Conf. Internet and Multimedia Systems and Applications* (IMSA 01), ACTA Press, 2001, pp. 334–338; http://collaboratory.emsl.pnl.gov/presentations/papers/ELN.IMSA.final.pdf.

[7] J. Myers, A. Chappell, M. Elder, A. Geist, J. Schwidder, "Re-integrating the Research Record," Computing in Science and Engineering, May/June 2003; http://collaboratory.emsl.pnl.gov/presentations/papers/reintegrating.html

[8] J. Whitehead. "WebDAV: Versatile Collaboration Multiprotocol", *IEEE Internet Computing*, vol. 9, no. 1, Jan/Feb 2005, pp. 66-74. http://www.soe.ucsc.edu/~ejw/papers/dav-ic-2005-final.pdf

[9] DAV Searching and Locating (DASL) Home Page, http://www.webdav.org/dasl

[10] Dublin Core Home Page, http://dublincore.org/

[11] J. Myers, et al., "A Collaborative Informatics Infrastructure for Multi-scale Science," Challenges of Large Applications in Distributed Environments (CLADE) Workshop, June 7, 2004, Honolulu, HI, p 24-33 http://collaboratory.emsl.pnl.gov/sam/resources/clade_copyright.htm

[12] DAV Explorer Home Page, http://www.ics.uci.edu/~webdav/

[13] South River Technologies Home Page, http://www.southrivertech.com/index.php?pg=./products/webdrive/index

[14] Apache Lucene Home Page, http://lucene.apache.org/java/docs/index.html

[15] G. Chin, C. Lansing. "Capturing and supporting contexts for scientific data sharing via the biological sciences collaboratory", *Computer Supported Cooperative Work, Proceedings of the 2004 ACM conference on Computer supported cooperative work*, 2004, pp. 409-418