

- Source Beamline 7. Technocal Report LBNL. Available at <http://gizmo.lbl.gov/opm.html>
- Edelson, D.C., Pea, R.D., Gomez, L.M. The Collaboratory Notebook. *Communications of the ACM*, 39(4): 32-33, April 1996.
- Erickson, T. The Design and Long-Term Use of a Personal Electronic Notebook: A Reflective Analysis. In: *Proceedings of CHI'96*, Vancouver, CA, April 13-18, 1996. ACM Press, pp. 11-18.
- Fowler, J., Baker, D.G., Dargahi, R., Kouramajian, V., Gilson, H., Long, K.B., Petermann, C., and Gorry, A.G. Experience with the Virtual Notebook System: Abstraction in Hypertext. *Proceedings of the CSCW'94*, 1994. ACM Press, pp. 133-143.
- Gwizdzka, J., Louie, J., and Fox, M.S. EEN: A Pen-Based Electronic Notebooks for Unintrusive Acquisition of Engineering Design Knowledge. In: *Proceedings of WET ICE'96*, Stanford, June 19-21, 1996. IEEE Computer Society, pp. 40-46.
- Halasz, F., Moran, T.P., and Trigg, R.H. NoteCards in a Nutshell. *Proceedings of CHI'87*, 1987. ACM Press, pp. 45-53.
- Kouzes, R.T., Myers, J., and Wulf, W.A. Collaboratories: Doing Science on the Internet. *Computer*, August 1996, pp. 40-46.
- Markowitz, V.M., Chen, I-M.A., Kosky, A., Kuno, H. and Szeto, E. 1996. Electronic Notebooks in the Context of the Object-Protocol Model (OPM) Tools. Working Document. August.
- Myers, J.D., Fox-Dobbs, C., Laird, J., Le, D., Reich, D., and Curtz, T. Electronic Laboratory Notebooks for Collaborative Research. In: *Proceedings of WET ICE'96*, Stanford, June 19-21, 1996. IEEE Computer Society, pp. 47-51.
- Neuwirth, C. Kaufer, D., Chimera, R. and Gillespie, T. The Notes Program: A Hypertext Application for Writing from Source Texts: Experiences and Writing. In: *Proceedings of the Hypertext'87 Conference*, 1987. ACM Press, pp. 121-141.
- Sachs, S.R., Agarwal, D., Tonner, B., Rotenberg, E., Denlinger, J. and Zhang, J. Requirements of a Notebook for Remote Collaboration. LBNL Internal Report, July 1995. <http://www-itg.lbl.gov/AWG.hm.pg.docs/notebook.req.html>; see also <http://www-itg.lbl.gov/~ssachs/notebook/report.html>
- Sachs, S.R. and Markowitz, V. 1996. Electronic Notebooks for Distributed Collaborators, research proposal submitted to the DCEE RFP, December 1995. LBNL Internal Report, February 1996.
- Sachs, S.R. and Myers, J.D. Working Group Report on Electronic Notebooks. *Proceedings of the 5th Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*. IEEE, 1996. pp. 53-58.
- Schnase, J.L., and Leggett, J.J. Computational Hypertext in Biological Modeling Applications. *Proceedings of the Hypertext'89 Conference*, 1989. ACM Press, pp. 129-135.
- Shipman, F.M., Chaney, R.J., and Gorry, G.A. Distributed Hypertext for Collaborative Research: The Virtual Notebook System. *Proceedings of the Hypertext'89 Conference*, 1989. ACM Press, pp. 129-135.
- Wulf, W.A. The Collaboratory Opportunity. *Science*, 261:854-855, 13 August 1993.

Acknowledgments

We gratefully acknowledge Brian Tonner, Deb Agarwal, Eli Rotenberg, Jonathan Delinger, Rick Steele, Al Thompson, and Werner Meyer-Ilse for the invaluable suggestions regarding electronic notebooks.

query window where icons representing the various classes (e.g., experiments, procedures, annotations, scans, devices, plots, and pictures) are displayed. Clicking on one of these icons shows a pop-up window with all the attributes of the selected class, and with a class hierarchy navigator starting from this class. As with the current visual query, selecting a specific attribute allows the entry of conditions on the values of that attribute. If the attributes are not of primitive types like integer, float or character, an identical window pops up to allow the specification of conditions over the attributes of that class. Each new condition that is specified can be composed with *and*, *or*, *not* operators. While creating a query, the user must have feedback on the query s/he is constructing. This is done by showing in the **query** window a sequence of icons, each representing a selected class and labelled with the specified conditions. The sequence of icons appears connected by the chosen *and*, *or*, *not* operators.

The results of a query are the objects that satisfy the conditions, all shown in a **results** window. The objects are represented by icons labeled with the objects identifiers. At this point, the user can click at the icons to see the information (text, picture, plots, etc.) associated with the retrieved objects.

7 Final Comments

We have presented the architecture and a prototype interface of the Spectro-Microscopy Electronic Notebook. The prototype electronic notebook is currently being evaluated by users with regard to its functionalities as compared to those of a paper notebook. Specifically, users are addressing the questions: (i) What can I do with the paper notebook that is not available with the electronic notebook? (ii) How much easier/difficult it is to use an electronic notebook to enter information as compared to a paper notebook? (iii) What can I do with the electronic notebook that cannot be done with the paper notebook?

At this point, the EN appears to allow all the functionalities that physicists have with their paper notebook. Although not classified as a difficulty of use, users have reported a need to change their style of use of a notebook, because the current EN inter-

face imposes a strict order in which information is entered. They have also reported that the electronic notebook considerably eases their task of adding annotation to past experiments, procedures, and projects, in addition to providing them with a history of the added annotations, including author and time of the update.

A question raised by users is: if changes are made to the database system or to the data acquisition engine, how are these changes reflected in the EN interface? This problem can be overcome by having the EN interface automatically built from the schema definition, and by extending the visual query language that we outlined in sections 5 and 6 to cope with the update of the database. This solution requires that users change the database schema to include the new parameters. This requirement, however, can be achieved with a minimum effort by the users. The automatic construction of the interface from the database schema definition requires the storage of some “presentation” parameters as an OPM class, in order to let some flexibility in the interface. Alternatively, a class “interface” could be created in the database through which users can specify the complete user interface.

As this prototype is evaluated, we expect that many other questions will be raised and we expect to have the opportunity to work on their answers!

References

- Assimacopoulos, A., Revillard, C., Herrmann, F., Borst, F., Paschoud, N., and Scherrer, J.R. An Electronic Notebook for Problem Oriented Patient Progress Notes: Testing a Concept. Proceedings of the 6th Annual Conference on Medical Informatics, 1989. pp. 813-817.
- Chen, I-M. A. and Markowitz, D. 1995a. An Overview of the Object Protocol Model (OPM) and the OPM Data Management Tools. Information Systems, 20(5):393-418.
- Chen, I-M. A., Kuno, H.A., Freitas, C.M.D.S., Markowitz, V.M., and Sachs, S.R. 1996b. The OPM Schema for the Spectro-Microscopy Collaboratory at Advanced Light

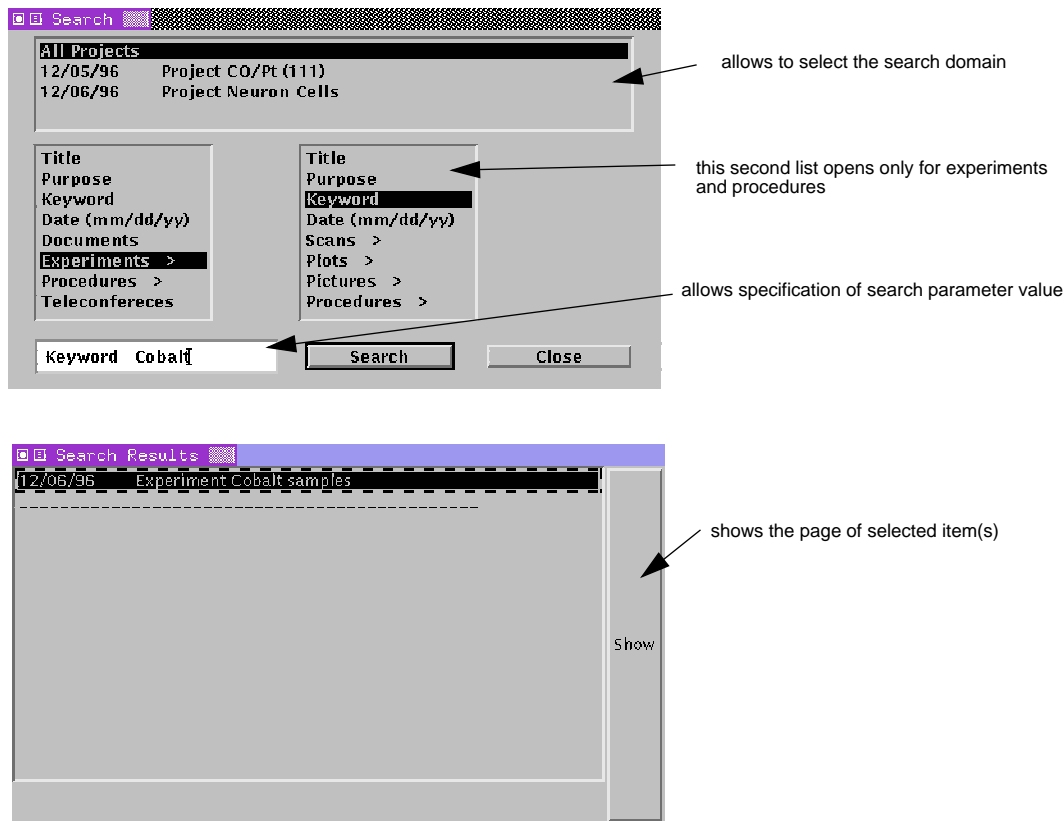


Figure 8. Search panel and results list allowing to show the pages of the retrieved items.

specify the appropriate parameters for the search. Specifying elements to be retrieved and the conditions for retrieval is accomplished by selecting from lists of classes and attributes, selecting a condition attribute, and filling out a text field with conditions. By pressing the **submit** button, OPM queries are issued that retrieve the instances that match the given constraints.

6 A visual query language for the Electronic Notebook

A visual query language is simply a query language where the query constructs are presented to the users in a visual form. A visual query language may encompass iconic expressions, diagrams and forms representation. The icon-based interaction provides

an easy way to select the desired classes of objects from a database, while the forms-based interaction allows the input of constraints on the selected set of instances of a given class.

In the previous section, we presented the visual constructs that can be used when simple searches are required. We focus here on the design of visual constructs for building complex queries.

The query functionalities presented in the last section do not allow for complex queries, because only one class/attribute can be selected for each search interaction, and only instances within the selected class are retrieved. The search for specific objects is done by selecting a class and specifying constraints on attribute values in order to retrieve the instances that satisfy the query conditions.

In order to support complex queries, we envision a

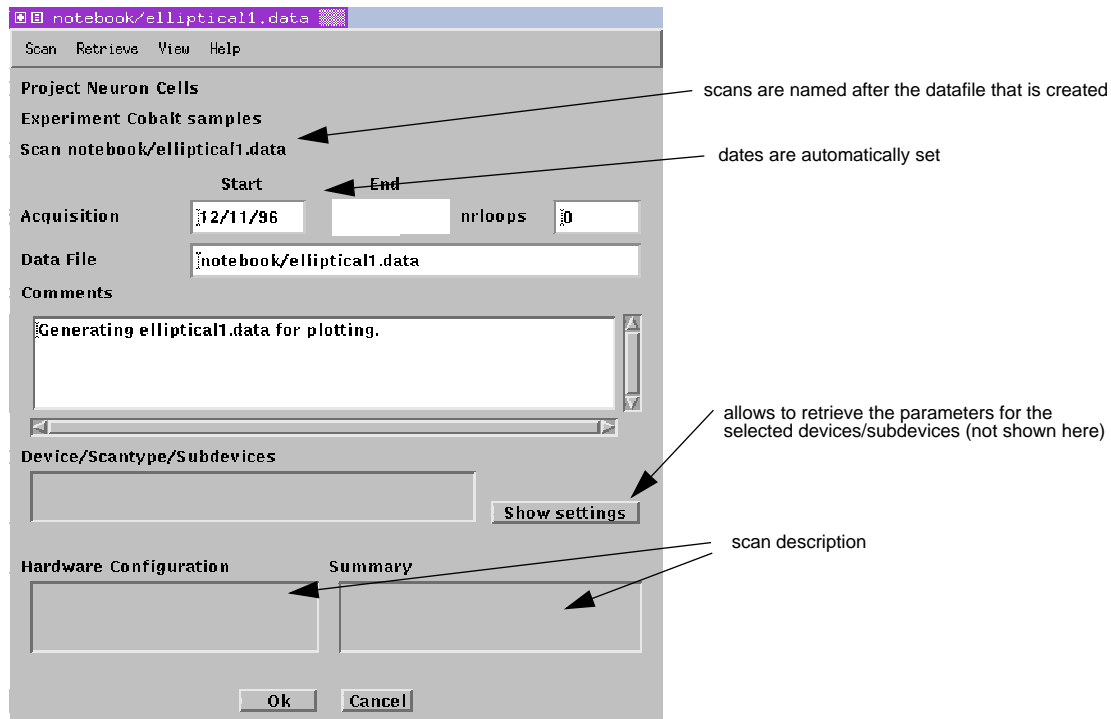


Figure 7. Page containing a scan description.

the user for title and comments on that scan. As was mentioned before, in this version of the electronic notebook, a data acquisition engine executes the scan, so that all the parameters are entered through that engine's interface and forwarded to the EN.

Annotations, plots, and pictures can be created and browsed through their specific pages. Both the **new+annotation** and the **retrieve+annotation** selection in the menu bar of **experiment page** leads to the presentation of one or more **annotation pages** that show a text and/or a button that invokes a drawing tool. The text can be edited; clicking on the drawing tool button, an existing drawing can be edited, or a new one can be created. The user can turn existing annotations pages, or can create a new one. To create a new annotation, a blank page with an empty text field and the button for the drawing tool is shown as the last page of the sequence.

Plots and pictures are created by using commercial packages or programs developed by the users. So, to allow the use of any tools, the concept of script was incorporated into plot and picture objects. Scripts are commands used to invoke (and provide settings to) the plotting tools mentioned above. Existing plots associated with experiments or scans

are shown as a list in the **plots page**. By clicking on an element of the list, the user can access a specific plot. In this page, the user can create a new script or create a new plot, using an existing script. When the user selects a new script, a scripting tool is started, and a new script can be created. A new plot is created by using the existing scripts; they are shown in a list, and the user can select and start the desired one, submitting data to it. If a plot is saved to a file, the user can enter the file name into the plot page, for future retrieval. In a very similar way, pictures can be retrieved from a page that shows them in a list form. The input of a new picture in the notebook requires only typing the name of the file that contains the picture, when no script was used to build it.

Instead of browsing experiments, procedures, scans, annotations, and other objects, the user can search for specific ones based on keywords (contained in all text fields of the database), or based on conditions established for attributes. A search can be started either through the **current projects page**, or through the **project page**.

The **search** button shows up a page (Figure 8) where the user can set the context of the search (all projects, only one project, or multiple projects) and

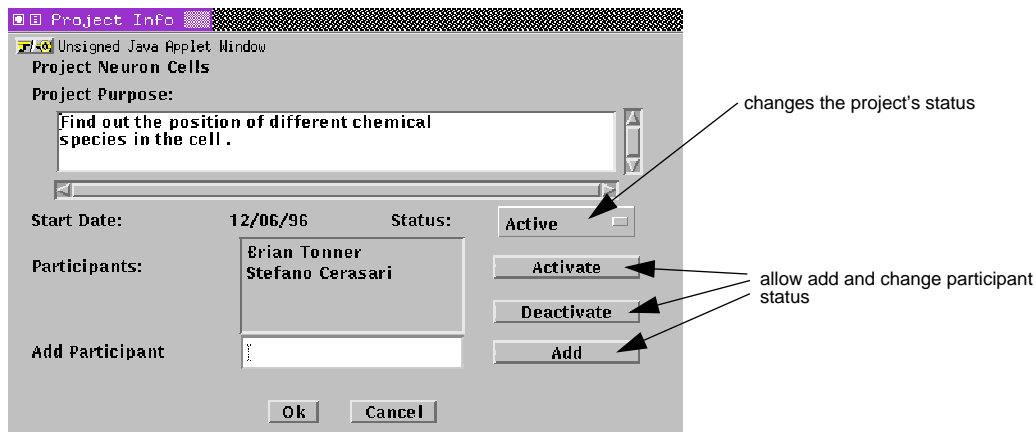


Figure 6. Project information page.

In an active project, all the participants can create new experiments, procedures, annotations, documents, and record teleconferences. They can also search and browse information already stored in the electronic notebook database. The update of all information related to a project is done by first browsing or searching the information, and then changing it. The same occurs with data acquisition or analysis; firstly, users access information about the project, experiment or procedures, and then when the user launches the data acquisition or data analysis engines, the produced scans, procedures, plots, and images are associated with the currently accessed database object.

A list of experiments and procedures, ordered by date or by title which can be set by the user, appears in the **project page**, so that the user can browse through the list to locate a particular experiment or procedure. By selecting one element of the list and pressing the **show** button, the user reaches the pages that describe it. The **experiment page** contains links to related procedures and scans. As in the project page, the menu on the top allows the user to start a new scan or a new procedure, create annotations, plots and pictures, and retrieve experiment general information (**experiment** option). Experiment information is shown as an **experiment info page**. The **retrieve** option provides for retrieving experiment's annotations, plots, and pictures. The **experiment info page** is also shown when the user

creates a new experiment (the text fields can be edited). For a new experiment, the form associated with this page must be filled. The **procedure page** and the **procedure info page** are very similar to experiment pages, thus they are not presented here.

The **experiment page** and the **procedure page** contain a list of procedures and scans, ordered by date or title, depending on the choice made by the user. The user can view detailed information about scans and procedures by selecting one of them from the list, and pressing the **show** button. This leads to the correspondent **scan page** or **procedure page**. **Scan pages** (Figure 7) show information about the selected scan, like parameters' settings, and datafile name. Under the passive mode user interface, new scans and procedures can only be launched via the data acquisition engine. However, the user should create an electronic notebook entry for new scans and procedures by selecting the **experiment** option, and from there, the **new+scan** option. By doing so, when a scan or procedure completes and the data acquisition engine notifies the EN engine, the EN engine can correctly “acquire” the scan/procedure information. Once the scan/procedure information is acquired by the EN engine, it will be reflected in the experiment page, in the scan/procedure page, and in the outline page.

A new scan can also be created from the **scan page** by selecting the **new** option. A text field prompts

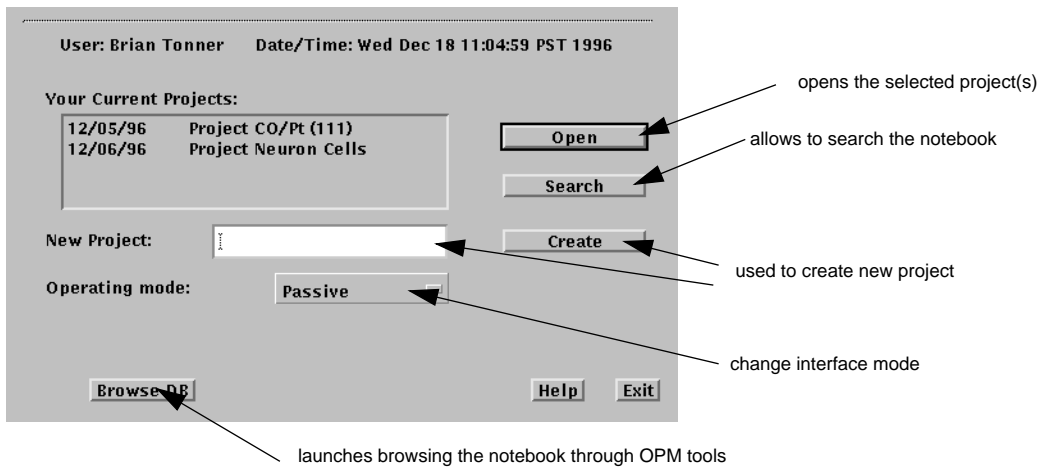


Figure 4. Current project page showing the projects owned by the current user.

The **project info page** (Figure 6) contains project title, project purpose, start and end date, list of participants, and current project status (active, suspended, terminated). This page also allows the project's principal investigator (owner) to modify the list of participants, project's purpose and status. A new project must have this page filled up by the owner. Through the **current projects page**, the user starts a new project by typing its title and issu-

ing the command **create**. Starting a new project implies entering project purpose and participants by filling up the correspondent text fields in the **project info page**. The starting date is automatically obtained from the system; also, the project becomes active. Once all the information in the project page is entered, the project is created in the database, and a **project page** for it is made available.

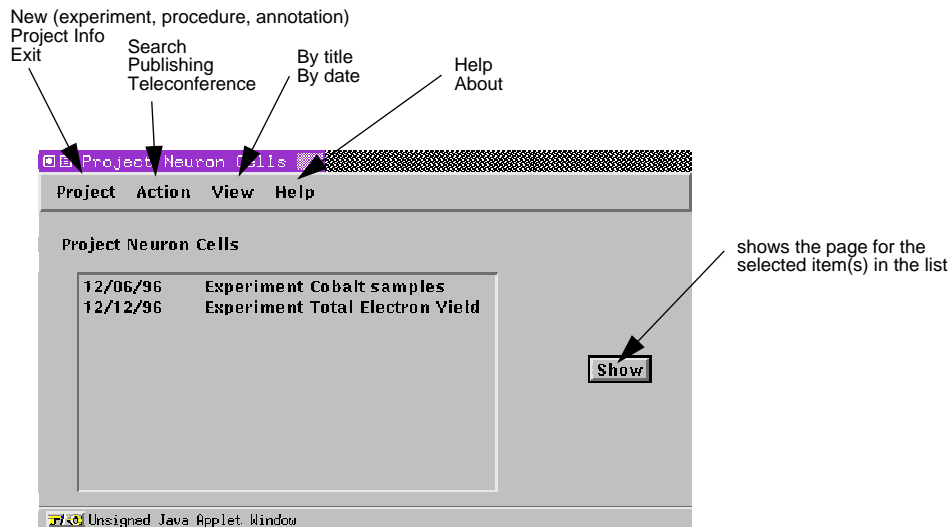


Figure 5. Page of a project showing a list of experiments.

Mode EN user interface.

After consulting with the users of our prototype, it became clear that the integration of the various user interfaces is not always desirable. So, we created the Passive Mode EN user interface, in which the information related to data acquisition, analysis and visualization is not directly obtained from the user via the EN interface. Instead, these engines communicate directly with the EN engine, and the passive mode interface simply reflects the data communicated to the EN engine by the other engines. The characteristic of this mode is an outline, organized by date, of actions generated by the EN engine and by all the other engines.

A Manual Mode was also determined necessary for the cases in which the users want to enter information about experiments in the same way they enter the information today on paper notebooks: simple, unstructured files. For these users, the automatic logging of data acquisition, or the recording of data analysis results are not available. If a user wishes to enter such information into the notebook, he/she must provide the names of created files.

It is also possible that the data acquisition, data analysis, and visualization engines cannot be modified to communicate with the EN engine. In these cases, the user must enter via the EN user interface the information about the location of files that are created by these engines, and he/she must associate them with a project or experiment. A default association can be used that associates the information manually entered with the currently opened project or experiment of the Electronic Notebook (with a confirmation and/or warning messages, to prevent errors).

A common understanding is that Electronic Notebooks must be remotely accessed and shared among collaborating researchers. To comply with this, the EN is launched within a Web browser like Netscape Navigator. The basic representations used to communicate with users are Web pages and Java-based panels.

The EN interface doesn't follow a single user interface paradigm. Instead, iconic, text, and forms are used providing a hybrid representation of data and commands. The first page provides to the user a brief general description of the EN, and presents the

login and password prompts. Only registered users are allowed to operate the electronic notebook.

Once allowed to enter the system, the **current projects page** (Figure 4) shows up, listing all the projects that the user currently participates in. The **current projects page** shows also the current user, date, and time, allowing the user to select a project or create a new one. As was mentioned before, different users have different preferences in operating the notebook. These preferences refer not only to how information is presented but also to the relationship among the notebook and the other engines in the system. To comply with this need, the **current projects page** allows the selection of the notebook's operating mode. In the following paragraphs the passive mode is described.

Suppose the user selects a project from those listed in **current projects page**: the electronic notebook presents the user with the **project page** (Figure 5), which contains a list of experiments and procedures developed in that project so far. This list is organized either in chronological order or in alphabetical order, depending on the current setting of the **view** option in the menu bar. The menu bar **project** option allows the user to start new experiments, procedures, and annotations, and to retrieve information about the project through a **project info page**. The **action** menu option provides for creating documents and recording teleconferences, as well as starting a search within the database. A **show** button is used in conjunction with the selection of an experiment or a procedure from the list. A specific page for the selected object is presented, as will be further detailed in this section.

If the user creates a new project, or chooses to work on a particular project (by creating new experiments or procedures), an **Outline page** appears. The Outline page of an existing project contains the Project title and an outline, organized by date, of all actions related to the project such as experiments, procedures, scans, etc. If the user creates a new project by using the **current projects page**, the new project title and date appears in the outline page. Similarly, if a new scan is started via the data acquisition engine, scan information appears in the outline indented with relation to the last listed item of the outline.

model actions that do not involve taking real data, even though scans may be used. Moreover, procedures can be composed actions, i.e. they can be composed of subprocedures, which are ultimately composed of a description and eventually scans.

Each scan corresponds to the data taken during a specific period of time by equipment attached to the BL-7. The class **scan** comprises the identification of the devices used in a particular scan, the starting and ending time when the measurements were taken, the configuration parameters for the devices, and the name of the data file generated by the data acquisition engine.

The class **Annotation** facilitates the association of textual and schematic information to an experiment. This class holds the information that physicists used to write in paper notebooks. The schematic information points to a file that stores a drawing made with a standard drawing tool.

The class **Plot** can be used to point to a graphic file that keeps some interesting plot, or represents the script that tells how to obtain a plot from one (or more) data files. It also holds information entered

by the user related to what are the significant points or regions in this plot. The class **Picture** represents images either obtained by a camera or computed by some tool. The picture is kept in a file, and this class points to it. The user can also enter information about significant points or regions in the picture.

5 The Electronic Notebook interface

A major requirement of Electronic Notebooks is that they should resemble paper notebooks. With paper notebooks, however, the question of how to access information is not discussed, because there are no alternatives: in order to locate information the user must browse through the notebook until he/she spots the desired information. As electronic notebooks bring several different alternatives to access stored information, different users have different preferences regarding the mode of accessing information. Initially, we designed an interface that integrated access not only to the Electronic Notebook, but also to the data acquisition, data analysis, and visualization engines. We named it Active

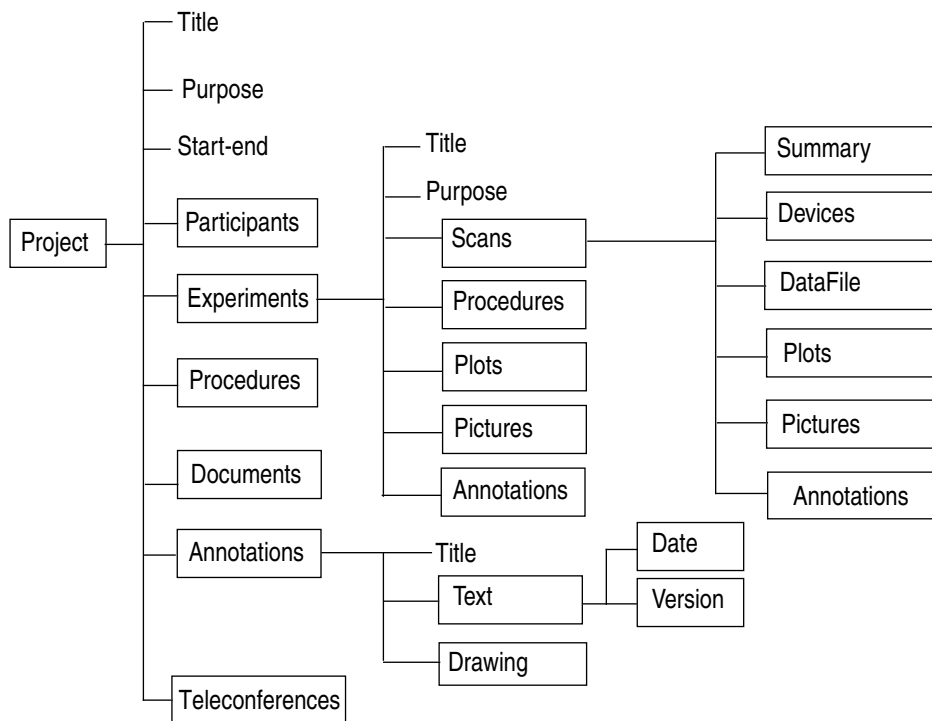


Figure 3. The main classes in the schema for the EN database. Not all attributes are shown.

- 3) selection of data presentation (for example, as tables or plots).

The **data entry** functions provide for the acquisition from local and remote users, and from the data acquisition, data analysis, and visualization engines. These functions were listed in the previous section under electronic notebook capabilities 1, 2, 4, 5, 6, and 7. It should be noted that the EN notebook doesn't need to provide all these functions by itself, but it must deal with the data captured or entered through a set of tools.

The **query** functionalities are related to browsing and retrieving information already stored in the database. This means the retrieval of all data based on the specification of a variety of conditions, using visual or textual formats. The textual format is a sentence written in the OPM Query Language. The visual format includes a simple interaction that combines menus with minimal text input, and a complex interaction that combines menus, textual and graphical representation of the data, and text input through forms.

OPM tools allow querying of the database either by using textual sentences through the OPM Query Translator or graphical interface using a schema-based browsing/querying tool. This graphical interface is generic, allowing the user to navigate through the hierarchy of classes of the schema, and to retrieve instances of the selected ones. The EN user interface provides its own specific constructs for visual queries, that are tailored to the specific needs of

physicists at ALS. However, the user can invoke the OPM browser tool within the EN interface.

4 The EN database schema

The Electronic Notebook database engine extends OPM tools for Electronic Notebooks, as described in [Markowitz et al. 96]. The schema is written in a textual description language for later translation into a specific database configuration. This language comprises constructs to declare classes, attributes, constraints on numbers of instances, and constraints on values of attributes. A construct to enter descriptions (comments) about the declared items is also provided.

The complete definition of the Spectro-Microscopy Electronic Notebook database schema can be found in [Chen et al. 96]. In the following, we give a general overview of the schema, presenting it through a very simplified diagram (Figure 3) and comments.

The higher level object in the database hierarchy is the **project**. A **project** carries management information and points to objects that contain descriptive information about experiments and procedures performed by physicists. An experiment is used to model actions that culminate in real measurements. These are taken by scanning samples put in the beamline chamber. An **experiment** is mainly composed of **scans**, but also has **procedures**, **annotations**, **plots**, and **pictures**. **Procedures** are used to

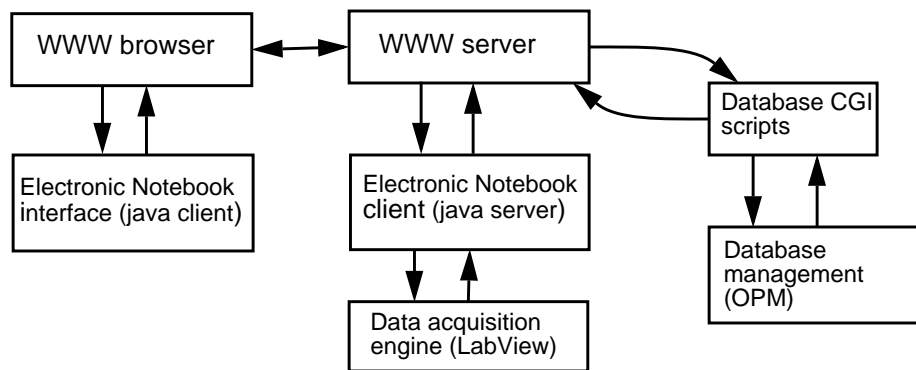


Figure 2. Current architecture of the Spectro-Microscopy Electronic Notebook.

ally displaying. It's essential that collaborators receive all updates that are being made by others while they are looking at the same data. So, the EN engine must selectively broadcast all the changes to the information stored in the electronic notebook.

The Database management engine is an extended set of the Object-Protocol Model (OPM) and the OPM tools [Chen and Markowitz 95] that offer uniform access to multiple databases systems and that can respond to EN engine requests. The Database management engine is fully described in [Markowitz et al. 96]. The EN engine was designed to submit commands to the Database management engine through the OPM CORBA interface. Command results are returned to the EN engine, which in turn passes them to the EN interface.

The Data acquisition engine collects data from the experiments via a LabView system, which has user interface based on control panels. This engine communicates to the EN engine the address of the collected data. Upon receiving this information, the EN engine converts the collected data into electronic notebook entries which are then available for browsing and searching. In this way, the Data acquisition engine operates independently of the EN user interface. As soon as acquired data is entered in the electronic notebook, it is made available to all the collaborators which can use the EN user interface tools for annotating, plotting, and analysis purposes.

The Data analysis engine is required to handle data analysis tools, collecting their input and output and writing them to the messaging engine, which in turn passes them to the EN engine. This engine is invoked by the user through the EN interface, and provides an interface between the EN engine and the variety of programs that users might want to use in data analysis tasks. Although many data analysis programs/tools produce numerical data, it is possible that some programs generate images as results. These can be stored in the electronic notebook and associated with the data used to produce it.

The Visualization engine has facilities for generating plots and pictures from data stored either as objects in the database or as separate data files, which are pointed to by database objects. This engine is envisioned as a set of built-in functions in the elec-

tronic notebook, supporting future tracking and steering capabilities, as well as visual representations of search results. Using the features of this engine, the user should be able to specify which visual representations are to be used for monitoring and controlling data acquisition, for example, and generate plots for analysis and publication purposes.

Currently, we are developing a prototype of the Spectro-Microscopy Electronic Notebook. The prototype has a Java-based Web interface, which invokes the database management engine by running a CGI (Common Gateway Interface) script at the database server site. Database management tools, such as browsing and querying, are also accessed in a similar way. This prototype does not yet implement the complete design because we have found too many problems with the current release of Java-to-Java communication. The architecture of the current prototype is shown in Figure 2.

Future plans for this prototype include:

- 1) the access of the database management engine via a CORBA *bus* and using Java networking capabilities (including multicast communications);
- 2) the automatic generation of customized electronic notebook user interfaces;
- 3) the communication of the EN engine to the EN interface using Java packets, and via a CORBA *bus*.

The development of the EN user interface and engine was based on the notebook functionality which can be tracked back to the list of needs mentioned in a previous report [Sachs et al. 95]. From the analysis of that list we can classify the functionality that should be provided by the EN into three classes: **control**, **data entry**, and **query**.

The **control** functions are intended to provide:

- 1) access to the EN by authorized users only. Moreover, it is necessary to provide different security levels;
- 2) automated recording of operations log, including the parameters' changes. This control function might be a simple *checkbox* that allows starting/stopping the recording of the log. All the information should be recorded in the database with author and time stamp;

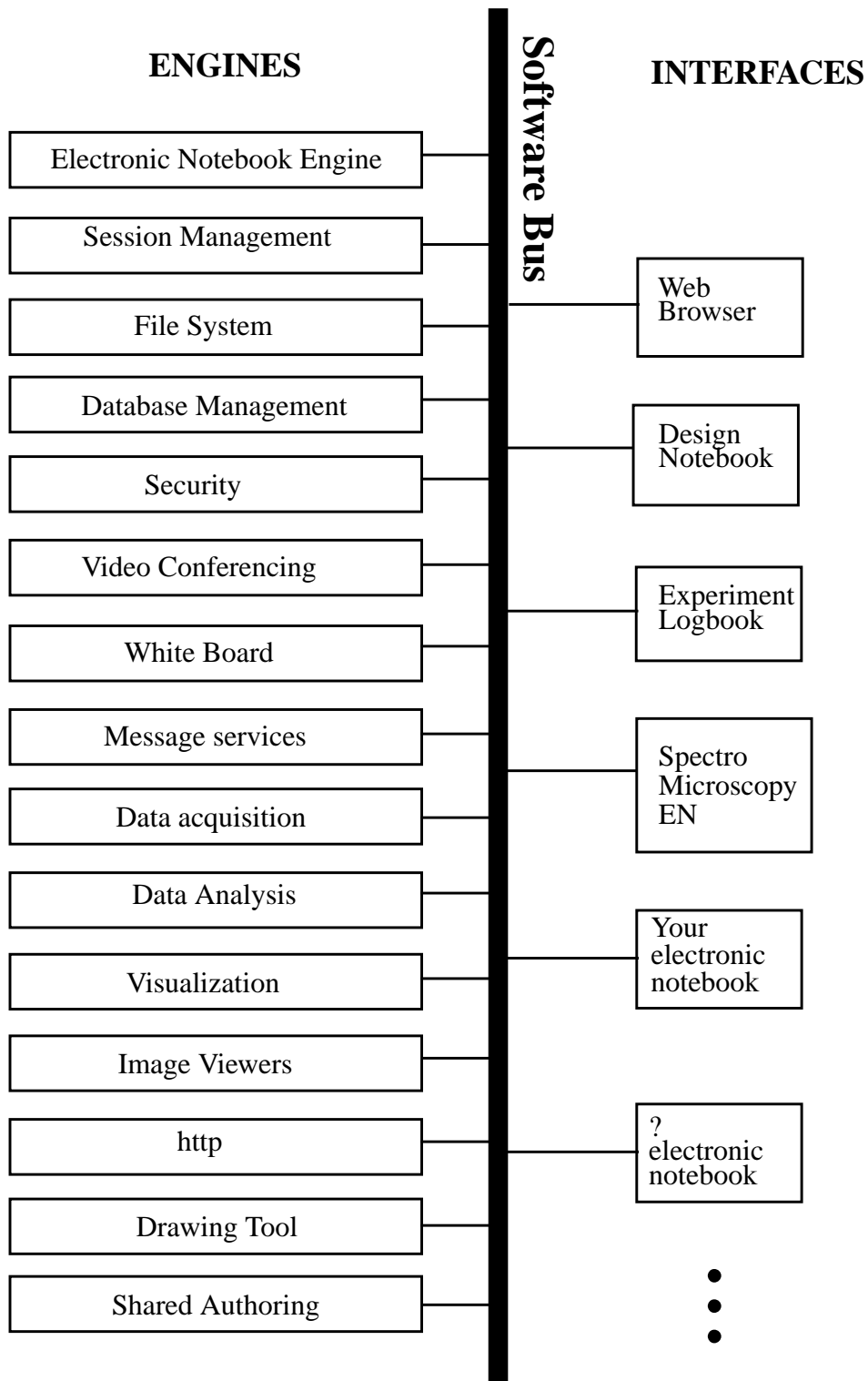


Figure 1. Architecture of Electronic Notebooks

book, providing full support for browsing, searching, and visualizing the stored data. The EN

engine communicates with all the clients (user interfaces) updating the information they are eventu-

erarchy of notebook and folders directories. Folders contain data about experiments and can be automatically generated by data acquisition programs. Experiment data include datafiles, instrument settings, operator comments, and user annotations. An image can be captured from the screen and linked to an annotation. Similarly, a file can be uploaded and linked to an annotation. The user interacts with the system through a Web browser; all the information stored in files is dynamically converted to HTML pages. Search results are also converted to HTML pages.

Another example of electronic notebook is the EEN [Gwizda, Louie, and Fox 96], an electronic engineering notebook mainly designed to acquire information and knowledge in engineering design processes. The system accepts handwritten text and sketches, as well as typed input. These objects can be dragged and dropped inside pages. Besides the data they represent, objects have several attributes, some automatically created by the system, others entered by the user through forms. Objects are stored in a persistent object storage and become part of a Knowledge Network that can be accessed through a Web interface. Two prototypes have been implemented, one in a desktop computer and the other using Apple Newton. The system focuses on acquiring design information using pen-based input and personal digital assistants. However, it relies only on user input and does not provide means for automatic data acquisition from other tools with the exception of a spreadsheet.

The collaborative electronic notebook architecture and implementation that are described in the following section intend to fulfill as many requirements as possible from the list mentioned earlier for a collaborative electronic notebook. The Spectro-Microscopy Electronic Notebook differs from the others in many characteristics. It is neither a hypertext application nor keeps the data in files. It is based on an object-oriented database accessed through a visual interface developed in Java, which is invoked using a Web-browser. However, no HTML forms or pages are created to allow the user to input data and to receive the output from queries to the database. The interface is mainly based on Java panels.

3 The Spectro-Microscopy Electronic Notebook Architecture

The architecture proposed for electronic notebooks at WETICE'96 [Sachs and Myers 96], as shown in Figure 1, consists of an electronic notebook *engine* that interacts with other *engines* and to notebook *user interfaces* via a CORBA-based *software bus*. The user interfaces shown in this Figure illustrate the fact that users may choose different interfaces to access their electronic notebooks. The electronic notebook engine receives requests from the users. In order to process the requests, it may request services from other engines. The electronic notebook engine may also receive requests from other engines, independently of the user interface.

Following this architecture, the Spectro-Microscopy Electronic Notebook (called EN, for short) encompasses a user interface designed specifically to meet the needs of physicists working with the Spectro-Microscopy facilities at the ALS, as well as several of the engines shown in the figure, namely, a electronic notebook engine, a database management engine, and data acquisition and visualization engines. Each one of these can rely on services provided by others.

The EN user interface was designed to allow physicists to enter and access (query, search, update) information into an electronic notebook as they remotely participate and conduct experiments. Information about all the projects in which they participate is available as soon as they connect to the system. Any updates made by any of the participating physicists is immediately available to all other participants. The EN user interface is described in Section 5.

The EN engine is a server for the EN user interface, and a client of the following engines: database management, data acquisition, visualization, and data analysis engines. Some of these engines do not have server capabilities, so that a messaging engine was designed to collect their output, and offer them to the EN engine. The design accounts for the possibility that other engines may be extended to offer server capabilities, so that the EN engine requests can be directly handled by them. The EN engine communicates with the database management engine in order to input or retrieve objects of the note-

might be separate notebooks with grant access facilities or a single collaborative electronic notebook. Some examples of collaborative electronic notebooks are Virtual Notebook System [Shipman, Chaney, and Gorry 89; Fowler et al. 94], CoVis notebook [Edelson, Pea, and Gomez 96], EMSL notebook [Myers et al. 96], ORNL Electronic Notebook Project (<http://www.epm.ornl.gov/~geist/java/applets/enote/>), and the EEN, an electronic engineering notebook [Gwizda, Louie, and Fox 96]. It is worth noting that some of the systems mentioned above were built to support specific tasks such as electronic design, or biochemical experiments.

The awareness that electronic notebooks are an important factor in the success of laboratories was developed as testbed laboratories were built and operated under the High Performance Computing and Communication (HPCC) initiative sponsored by the Department of Energy.

A notebook for laboratories may include the following capabilities:

- 1) Automated and manual entry of parameters used to configure the experimental facilities, including templates for formatted entry.
- 2) Automated creation of operations and equipment logs.
- 3) History of parameter changes.
- 4) Reference and annotation to results files, graphs of output data, and other experiments.
- 5) Entry of text with descriptions and discussions about the experiments (including entry of scientific characters and symbols).
- 6) Entry of sketches and drawings used to describe experiments, images captured from samples or processed from output data, sequences of images that can be played as movies, selected video/audio clips of teleconferencing, related computer files, graphic images, and paper reports.
- 7) Entry of cross-experiments annotations, discussions, and comparison of graphs, images, and sequence of images.
- 8) Indexing and retrieval methods of all experiment data, including annotations, plots, and images.

- 9) Integration of existing data analysis and visualization tools, including the capability of cutting and pasting graphs, figures, and images.

Electronic notebooks for laboratories are inherently shared among collaborating researchers or engineers. Secure access of electronic notebooks is a major issue because only authorized collaborators must be allowed access to them. Also, it is desirable that participants keep some of their annotations privately. Thus it is paramount that privacy mechanisms are included. Associated aspects are non-tamperability (write-once) of notebooks, and their use as patent records for inventions or as proof of conformity to procedures. Multiple notebooks may be necessary for each collaboration effort, and each researcher or engineer may need to access notebooks of different laboratories. Tools to coordinate access to multiple notebooks are needed.

Current efforts in electronic notebooks for laboratories try to fulfill the above requirements. Much progress and effort is still required in many areas such as input techniques, database technology, visualization techniques, and security, to name just a few.

Virtual Notebook System (VNS) [Shipman, Chaney, and Gorry 89; Fowler et al. 94] supports authoring and viewing of hypermedia documents. Users access their own notebooks as well as others' to which they have been granted access. The notebooks are stored in libraries, so users connect to a library before getting to the notebook they need. A VNS notebook is a group of pages; each page can have several types of objects: text, image, link, action, and graphic (drawing) objects. Link objects provide for connecting pages even located in different notebooks. Action objects allow the execution of external programs and there are functions for exporting and importing text and images.

While VNS is more generic, oriented to authoring of hypermedia documents, the EMSL notebook [Myers et al. 96] and ORNL Electronic Notebook Project (<http://www.epm.ornl.gov/~geist/java/applets/enote/>) are devoted to the recording of experiments conducted in laboratories.

The EMSL electronic notebook [Myers et al. 96] maintains a collection of flat files organized as a hi-

1 Introduction

The Spectro-Microscopy Electronic Notebook for the Advanced Light Source Beamline 7 (ALS-BL7) is a system that allows storage and retrieval of information generated as collaborating physicists run and analyze experiments at the ALS-BL7. The Advanced Light Source is the first low-energy third-generation synchrotron radiation source in the U.S., providing the brightest light in the ultraviolet and soft x-ray regions of the spectrum. In the several beam lines of the ALS, there is a variety of equipment and microscopes that take measurements from samples of materials in beamline chambers. The Spectro-Microscopy project is a collection of experiments designed to maximize the benefits of the undulator and monochromator on the ALS Beamline 7. The experiments combine high spatial resolution and high energy resolution for the study of chemical samples. The various equipment at the ALS -BL7 are used by a team of investigators from several academic and research institutions, thus presenting an environment with all the characteristics for housing a collaborative - a tool-oriented computing and communication environments that support scientific and engineering remote experimentation and collaboration¹.

The Spectro-Microscopy Collaboratory project has the goal of using current network and video-conferencing technology to provide remote access to these facilities. Collaborating physicists remotely monitor data acquisition and discuss experiment configuration and results. Data acquired during experiments are stored as text files, in a proprietary format. Data analysis tools are used to generate images (e.g., IDL) or plots (e.g. IGOR, Labview) from collected data. Tele-conferences facilities are provided to allow remote collaborators to communicate with local researchers during experiments. The recording of experiments is currently done by the physicists on paper notebooks, where they keep notes about parameters used to configure the equipment, the names of generated files, comments, diagrams, and conclusions, as well as plots and images related to the experiments. Paper note-

1. refer to [Wulf 93] [Kouzes, Myers, and Wulf 96] for further information about the origins of collaboratories.

books, however, are not adequate to record experiments conducted remotely by a team of physicists because the recorded information must be on-line and accessible by each team member. The Spectro-Microscopy Electronic Notebook project [Sachs and Markowitz 96] was created to achieve this goal.

This report presents an overview of the Spectro-Microscopy Electronic Notebook, addressing issues concerning its overall architecture and functionality and a first approach to its user interface. The text is organized as follows. Next section covers background and related work on electronic notebooks for collaboratories. Section 3 is a general overview of both the envisioned architecture of electronic notebooks and our current prototype. Section 4 briefly describes the schema for the electronic notebook database. Section 5 presents the current design of the electronic notebook user interface, while section 6 addresses the specific questions of a more powerful visual query language. The last section draws some comments on the first results of user's evaluation and questions that have already been raised.

2 Background and Related Work

Electronic notebooks are media in which scientists, engineers, and artists can record aspects of their observations, analysis, synthesized notes, and conclusions. It is also a medium for collaborative scientific inquiry or engineering design discussions.

Regarding the capabilities of sharing information, electronic notebooks can be classified as personal or collaborative. Personal electronic notebooks are intended to be used by a single user to input, store, organize, and manipulate personal data such as daily To Do lists, records of meetings, and thoughts. Some examples are Notes [Neuwirth et al. 87], NoteCards [Halasz et al. 87], a physician's notebook [Assimacopoulos et. al. 89], a biologist's notebook [Schnase and Legget 89], and Proteus [Erickson 96]. Collaborative electronic notebooks are intended to support collaborative work, providing similar functionalities. In such systems, collaborators are allowed to access common data, which can be annotated by each collaborator. There

Contents

1 Introduction 4

2 Background and Related Work 4

3 The Spectro-Microscopy Electronic Notebook Architecture 6

4 The EN database schema 9

5 The Electronic Notebook interface 10

6 A visual query language for the Electronic Notebook 14

7 Final Comments 16

References 16

Acknowledgments 17

Abstract

This paper gives an overview of the Electronic Notebook for the Spectro-Microscopy Collaboratory at the Advanced Light Source Beamline 7 (ALS-BL7). The Spectro-Microscopy Collaboratory project has the goal of using current network and video-conferencing technology to provide remote access to the facilities at ALS-BL7. The Electronic Notebook is a tool that allows physicists accessing the ALS-BL7 facilities to store and retrieve all information generated as they collaborate to run experiments. The Electronic Notebook replaces a multiplicity of manual and automated procedures currently used for storage/retrieval of data associated with experiments at the ALS-BL7. In addition, the Electronic Notebook offers new and powerful capabilities, while providing users with a homogeneous user interface to various tools. This paper outlines the architectural design of the Electronic Notebook, and describes its visual interface, which is used to prompt local and remote users to enter information related to their experiments, and provides query and browsing facilities to enable information retrieval.

The Spectro-Microscopy Electronic Notebook¹

Sonia R. Sachs²
Carla M. Dal Sasso Freitas³
Victor Markowitz⁴
Anna Talis⁵
I-Min A. Chen⁶
Ernest Szeto⁷
Harumi A. Kuno⁸

Joint work

Information and Computing Sciences Division
Lawrence Berkeley National Laboratory
1 Cyclotron Road, Berkeley CA 94720

and

International Computer Science Institute
1947 Center Street Suite 600
Berkeley CA 94704

January 1997

-
1. Issued as Technical Report LBNL-39886 and ICSI TR-97-002. This work is supported by the DCEE Program, U.S. Dept. of Energy, Office of Energy Research, Mathematical, Information and Computational Sciences Division.
 2. ICSD, Imaging and Distributed Computing group. SRSachs@lbl.gov.
 3. Visiting scholar at LBNL and International Computer Science Institute, Berkeley, CA on leave from Universidade Federal do Rio Grande do Sul, Porto Alegre, Brazil, CFreitas@lbl.gov, freitas@icsi.berkeley.edu, carla@inf.ufrgs.br. Grant CNPq/Brazil 201404/95-8.
 4. ICSD, Database management group. VMMarkowitz@lbl.gov
 5. ICSD, Imaging and Distributed Computing group, ATalis@lbl.gov
 6. ICSD, Database management group, IACHen@lbl.gov
 7. ICSD, Database management group, E_Szeto@lbl.gov
 8. Formerly at ICSD Database management group.