

Project management system for structural and functional proteomics: Sesame

Zsolt Zolnai^{1,*}, Peter T. Lee¹, Jing Li¹, Michael R. Chapman[†], Craig S. Newman¹, George N. Phillips Jr.¹, Ivan Rayment¹, Eldon L. Ulrich¹, Brian F. Volkman² & John L. Markley¹

¹Center for Eukaryotic Structural Genomics, Department of Biochemistry, University of Wisconsin-Madison, Madison Wisconsin 53706; ²Medical College of Wisconsin, Milwaukee Wisconsin 53226, USA

*Author for correspondence (e-mail: zsolt@nmrfam.wisc.edu; fax: +1 608-262-3759)

Received 16 July 2002; Accepted in final form 29 October 2002

Key words: data bank deposition; data harvesting; genomics; information management; proteomics; structure determination

Abstract

A computing infrastructure (Sesame) has been designed to manage and link individual steps in complex projects. Sesame is being developed to support a large-scale structural proteomics pilot project. When complete, the system is expected to manage all steps from target selection to data-bank deposition and report writing. We report here on the design criteria of the Sesame system and on results demonstrating successful achievement of the basic goals of its architecture. The Sesame software package, which follows the client/server paradigm, consists of a framework, which supports secure interactions among the three tiers of the system (the client, server, and database tiers), and application modules that carry out specific tasks. The framework utilizes industry standards. The client tier is written in Java2 and can be accessed anywhere through the Internet. All the development on the server tier is also carried out in Java2 so as to accommodate a wide variety of computer platforms. The database tier employs a commercial database management system. Each Sesame application module consists of a simple user interface in the client tier, corresponding objects in the server tier, and relevant data stored in the centralized database. For security, access to stored data is controlled by access privileges. The system facilitates both local and remote collaborations. Because users interact with the system using Java Web Start or through a web browser, access is limited only by the availability of an Internet connection. We describe several Sesame modules that have been developed to the point where they are being utilized routinely to support steps involved in structural and functional proteomics. This software is available to parties interested in using it and assisting to guide its further development.

Abbreviations: BMRB, BioMagResBank <<http://www.bmrwisc.edu>>; CESG, Center for Eukaryotic Structural Genomics <<http://www.uwstructuralgenomics.org>>; CGI, Common Gateway Interface; COG, Cluster of Orthologous Groups; CORBA, Common Object Request Broker Architecture; DBMS, DataBase Management System; DCOM, Distributed Component Object Model; GUI, Graphical User Interface; IDL, Interface Definition Language; IIOP, Internet Inter-ORB (Object Request Broker) Protocol; ISGO, International Structural Genomics Organization; JDBC, Java Database Connection; LIMS, Laboratory Information Management System; NMRFAM, National Magnetic Resonance Facility at Madison <<http://www.nmrfam.wisc.edu>>; ORB, Object Request Broker; ORF, Open Reading Frame; PDB, Protein Data Bank <<http://www.rcsb.org>>; RDBMS, Relational DataBase Management System; RMI, Remote Method Invocation; SG, Structural Genomics; SQL, Standard Query Language

[†]Deceased, 30 August 2000.

Introduction

The successes of the myriad genome sequencing projects are creating increasing demands for rapid enlargement of the database of information on protein sequence–structure–function relationships. The imbalance between the rates for determining gene sequences vs. protein structures points to the need for faster and more efficient methods for the latter. The two approaches for determining high-resolution protein structures are single-crystal diffraction analysis and solution-state NMR spectroscopy. Each approach has its own strengths and weaknesses, but together the two reinforce and complement one another. X-ray crystallography and NMR contribute about 85% and 15%, respectively, of the structures being deposited currently in the Protein Data Bank (PDB). Crystallography, which requires production of good-quality crystals, provides highly resolved snapshots of particular conformations compatible with the crystal lattice but little atomic-level information from regions that are statically or dynamically disordered. NMR, which requires protein solubility at the level of about 0.1 mmol/L, provides information at atomic resolution level for both structured and unstructured regions as well as insight into the chemical properties of individual sites under defined (variable) conditions.

Structure determination by either X-ray crystallography or NMR spectroscopy involves a large number of tasks and operations carried out on individual targets. Keeping track of the samples and the information concerning their production and analysis is a challenge, even for structural biology projects that typically involve a few protein targets. The usual methods are to keep annotations in laboratory notebooks and to use separate, uncoordinated software packages for data collection and processing. This approach no longer suffices for structure determinations on the scale demanded by structural proteomics projects.

Analysis of the information technology requirements for structural proteomics requires consideration of the various steps involved (Figure 1). The first steps are the selection of the target and the design of a strategy for producing the full-length cDNA. Next, methods for protein expression, detection, and isolation, are chosen and evaluated. The initial steps leading to the production of pure protein, those generally considered to constitute the major slow step or 'bottleneck' in structural proteomics [1], are identical for

NMR spectroscopy and X-ray crystallography. Thereafter, the procedures diverge.

With X-ray crystallography, the first step is the identification of conditions for producing suitable crystals; promising targets then must be heavy atom labeled. Suitable crystals are taken to a synchrotron, where data are collected under a variety of protocols. Each target may be represented by one or more crystals, depending on the phasing strategy to be employed. The time required per crystal is short, and it is possible to collect data on 25 or more crystals in one day.

With NMR, conditions must be found for optimal protein solubility and stability, promising protein targets then must be prepared with stable isotope enrichment to facilitate NMR signal detection and analysis. The time requirements for NMR data collection, while far greater for X-ray crystallography, have been reduced recently through the development of more-efficient pulse sequences and more sensitive NMR spectrometers employing cryogenic probes and preamplifiers. It is possible now to collect a full set of data for an NMR structure determination in 3–5 days. Thus, a dedicated NMR spectrometer should be capable of producing data sets for 50 or more protein structures/year.

The subsequent steps leading to structure determination, refinement, and validation, while somewhat different for X-ray and NMR, are highly dependent on software and data visualization. These steps offer the potential for appreciable improvements through automation. The final steps of data deposition and report writing require detailed information from all previous steps in the process.

Software tools for streamlining the steps in structural proteomics need to address four problems. First, the work typically involves different persons (or groups of persons) at different stages. All of these people require appropriate views of the project. Second, although software may exist for carrying out individual steps (in many cases different competing packages for a given step), incompatibilities in the input/output data prevent users from mixing and matching different programs. Third, because of the complexity of structural investigations, it is difficult to keep track of protocols used, the content, history, and current location of samples, and the storage locations of primary and intermediate data. Fourth, contrary to current trends in information technology, which allow for mobile services, steps in protein structure determi-

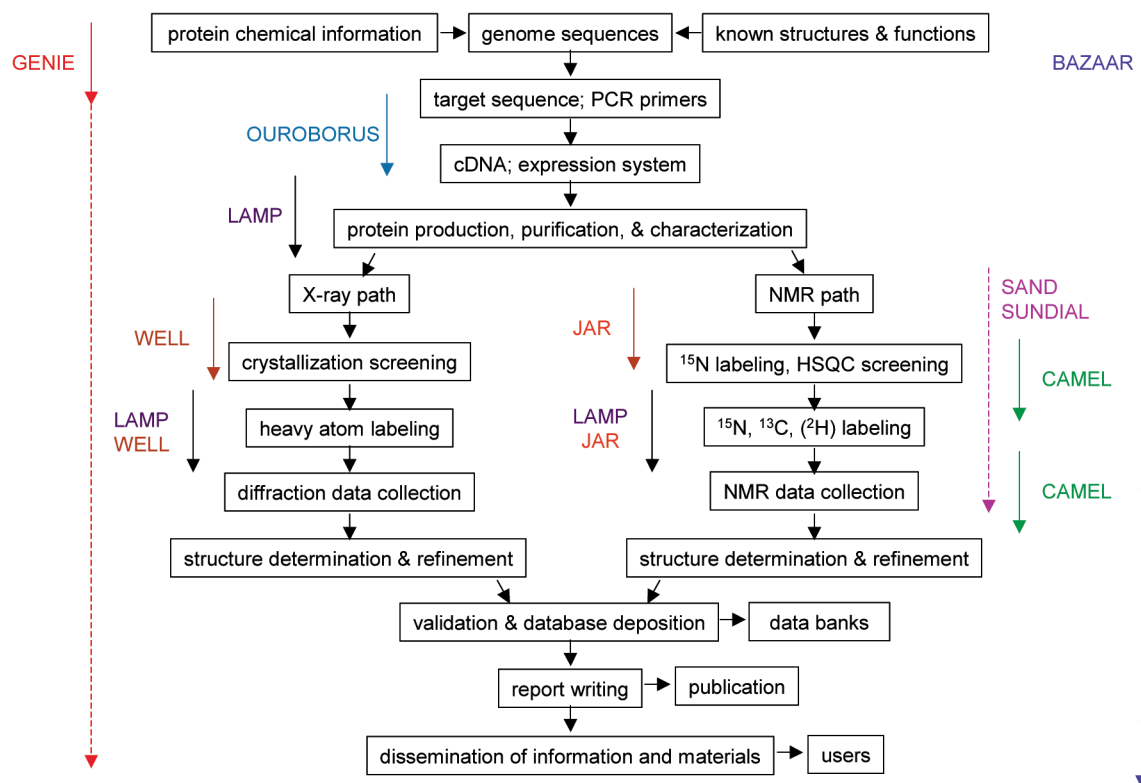


Figure 1. (Black boxes and arrows) Work and information flow within a structural proteomics project utilizing structure determination by both X-ray crystallography and NMR spectroscopy. (Colored items) Modules of the project management system (Sesame) developed to date that address portions of the work and information flow within a structural proteomics project. These Sesame modules are discussed in the Results Section of the text.

nations are largely tied to fixed sites and hardware. The reasons for this stem from limitations in current software, the large total data size (per project), and demands for rapid processing. For example, experiment setup, control, and view normally are tied to the computer, or class of computer, on which the data collection was initiated, making remote setup assistance, experiment stop/restart, or progress checking extremely cumbersome. Most of the processing stages also are non-mobile. Limiting factors include the large sizes of raw data sets, which prohibit invisible data transfer among computers on the network, and the large-scale computational resources required at particular stages in data processing.

The Center for Eukaryotic Structural Genomics (CESG) at the University of Wisconsin-Madison has chosen to address these problems by developing a project management system based on an earlier software package created at the National Magnetic Reso-

nance Facility at Madison (NMRFAM). This system is the Sesame project management system, which utilizes an open model for data input/output (for example database tables and public methods to manipulate them) and the multi-tier client/server paradigm. The Sesame system is designed to organize imported information, to assist in planning experiments to be carried out on single targets or groups of targets, to launch experiments and track their progress, and to collect, process and catalogue results. The goal is to provide views on the progress of individual structure determinations and longitudinal views of progress for entire projects and to facilitate report writing and database depositions. Sesame has the capability of interfacing with the variety of legacy software packages used in structure determinations so that information flowing into and out of these packages can be organized without the need for rewriting these packages.

Methods

General approach

The Sesame software package follows the client/server paradigm. It consists of a framework, which supports secure interactions between the three tiers (Figure 2) of the system (the client, server, and database tiers), and application modules, which carry out specific tasks (Figure 1). The framework utilizes industry standards. The client tier is written in Java2 and can be accessed anywhere through the Internet. All the development on the server tier is also carried out in Java2 so as to accommodate a wide variety of computer platforms. The database tier employs a commercial database management system.

Sesame makes use of the Object Web paradigm in which the object components of the client and server sides of an application can reside anywhere on the Web; they work together thanks to the Object Request Broker (ORB), which manages the communication between them. For communication between different ORBs the Internet Inter-ORB Protocol (IIOP) is used.

The Sesame objects describe the structural-genomics-related concepts. They are application-level components that are independent of any single application; they can be used in unpredictable combinations. A subset of Sesame data structures is shown in Figure 3. A Sesame application module consists of clusters of Sesame objects, and it simply provides the environment to execute them. The Sesame objects recognize events in their environment, change their attributes, interact with other Sesame objects, and can be reused in different application contexts.

A Sesame object component consists of three kinds of objects. (1) Master objects encapsulate the storage, metadata, and rules associated with an active Sesame object and its short-lived processes; they define how the object reacts to changes in the view or model. (2) Processing objects handle long-lived processes that involve one or more Sesame objects and define how the object reacts to changes in environment. (3) Presentation objects represent the object visually to the user and communicate directly with the master object and sometimes with the processing object.

Each Sesame object can have multiple presentations, which can be spread across multiple clients. Note that some Sesame objects may be entirely process oriented and not associated with any specific data or presentation. The processing objects may reside in one or more servers. Under the CORBA architecture, all objects have IDL-defined interfaces and connect to one another on the same computer or on different computers via ORBs (transparent scalability) [2].

Sesame objects usually are decomposable into smaller parts: components that can be reassembled by the three-tier client/server model. The first tier represents the visual aspects of the Sesame object, which usually reside on the client. The second tier comprises the server objects that represent the persistent data and the processing functions. The third tier comprises the Sesame database and legacy databases. The Sesame database also contains administrative tables that connect the component properties to users, experiments, and projects, and govern the access privileges. The server objects interact with their clients and with the databases, and implement the logic of the Sesame

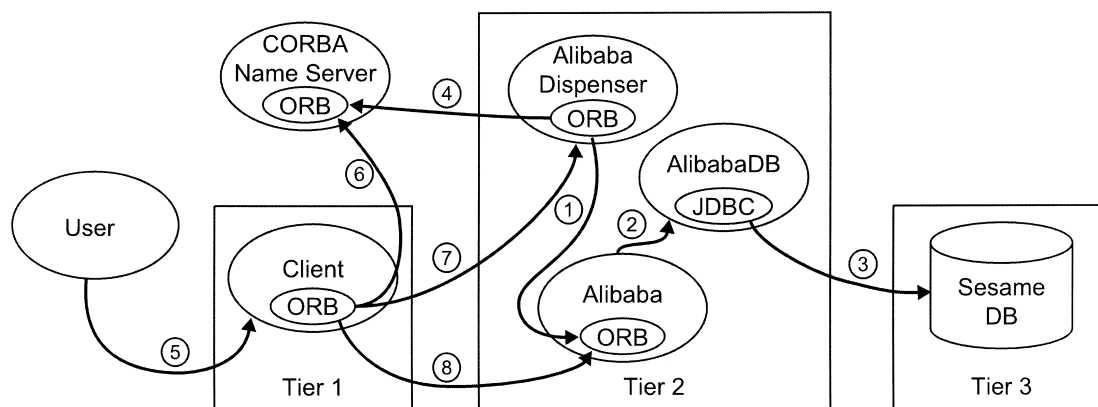


Figure 2. Three-tier organization of the Sesame project management system. The first tier consists of the user interface accessible via web browsers. The second tier carries out heavier computations. The third tier consists of the database.

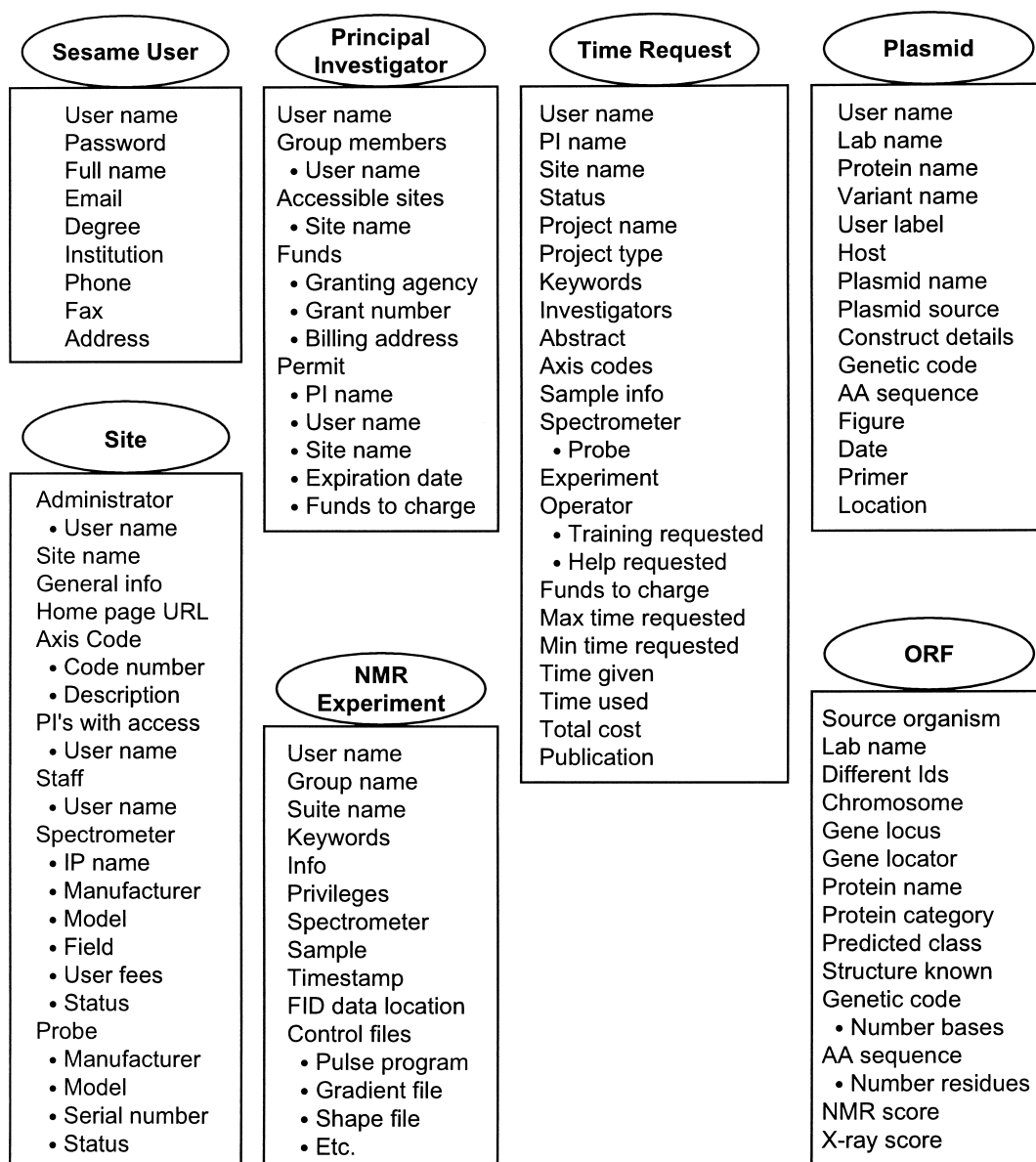


Figure 3. Representative data structures in Sesame.

object. The clients never interact directly with the third tier.

In developing Sesame, the software component model has been used to simplify code development and to maximize its reusability. Components are specialized self-contained software entities that can be customized and inserted into application modules. Interaction with components occurs through event handling and method invocation. The appearance or beha-

avior of a component is determined by its properties. Components can be either visible graphical user interface (GUI) components or invisible algorithmic components. Containers are components that contain other components. Containers are used in the visual organization of GUI components or in connecting algorithmic components into more complex entities that support complex scientific processing schemes. The use of containers and components supports hier-

archical software development and allows maximum reuse of existing software when creating new software or improving existing software. Another advantage of the component-based approach is that one can use visual design tools to simplify software development.

Sesame modules

Basic components are combined into more-complex components or modules, such as those with the functionality shown in Figure 1. Each module follows the three-tier architecture (Figure 2).

The first tier of the module is a light graphical user interface written in Java2 to assure platform independence. Its task is to present visual representations of objects in the second tier. The graphical interface allows the user to inspect, change, or load attributes from corresponding tables in the database (third tier).

The second tier of the module generally consists of multiple objects: one of them talks to the first tier client object, another to the database in the third tier, and others handle data processing and implement an additional security layer. The second-tier objects are also written in Java2 to assure platform independence. The behavior of the second tier objects is regulated by attributes controlled by the first tier, which are stored in the third. In addition to these objects, the second tier contains administrator objects that monitor the resources, and dispense new objects if needed.

The third tier of a module is a relational database, which consists of multiple tables containing the component attributes. In addition, the database contains administrative tables that govern privileges for data access and connect the component attributes to users, associations, experiments, and projects. The first and second tiers communicate by ORB, whereas the second tier uses JDBC to talk to the third tier. This approach makes it possible to track (via the database) all the steps carried out during a project. It also supports communication among collaborators (via privileged database access), development of custom interfaces and processing schemes (using available GUI and JavaBeans¹ utilities), and the introduction of new functionalities or extension of existing ones (via JavaBeans). Items in the database (in the third tier) that

correspond to those archived in the structural (PDB) and NMR spectroscopic (BMRB) databases are stored in formats that support easy data deposition. Figure 4 lists representative processes (accessible from various Sesame screens) that facilitate interaction with the database.

The Sesame infrastructure supports a large number of clients, contains servers that are specialized for various processing operations, and can interact with a number of databases in addition to its own relational data base management system (RDBMS).

Interfacing sesame to instrumentation

Software packages that control X-ray or NMR data acquisition, laboratory workstations, or laboratory robots (cloning, sample preparation, crystallization, etc.) typically are script or parameter file driven. Conversion of a script-driven application into a three-tier distributed component requires the development of GUI-based client components (presentation objects) to manipulate the parameters of the commands. By connecting these components, more complex components can be built that correspond to scripts and/or create the appropriate parameter files. The second tier contains objects that build particular scripts/files from the parameters selected by the first tier, along with objects that do the processing. The processing objects launch the corresponding script-driven programs with the scripts as arguments. These objects are 'quasi' objects, because they are Java2/CORBA-wrapped legacy programs. The third tier is the relational database, which contains the parameter (script and control) tables for programs and additional administrative information (user, group, spectrometer, etc.). The first (client) and third (database) tiers are designed in such a way that they require no (or minimal) change once the second tier has been replaced by 'true' objects. These objects can be acquired by one of three approaches: (1) if the source code is available, it can be modified to fit the Sesame (distributed three tier components) paradigm; (2) if the source code is unavailable, but the author/company is willing to modify it, instructions and wrappers can be supplied to the author/company; (3) otherwise new objects can be created.

¹JavaBeans are easily reusable classes that can be manipulated by visual development tools.

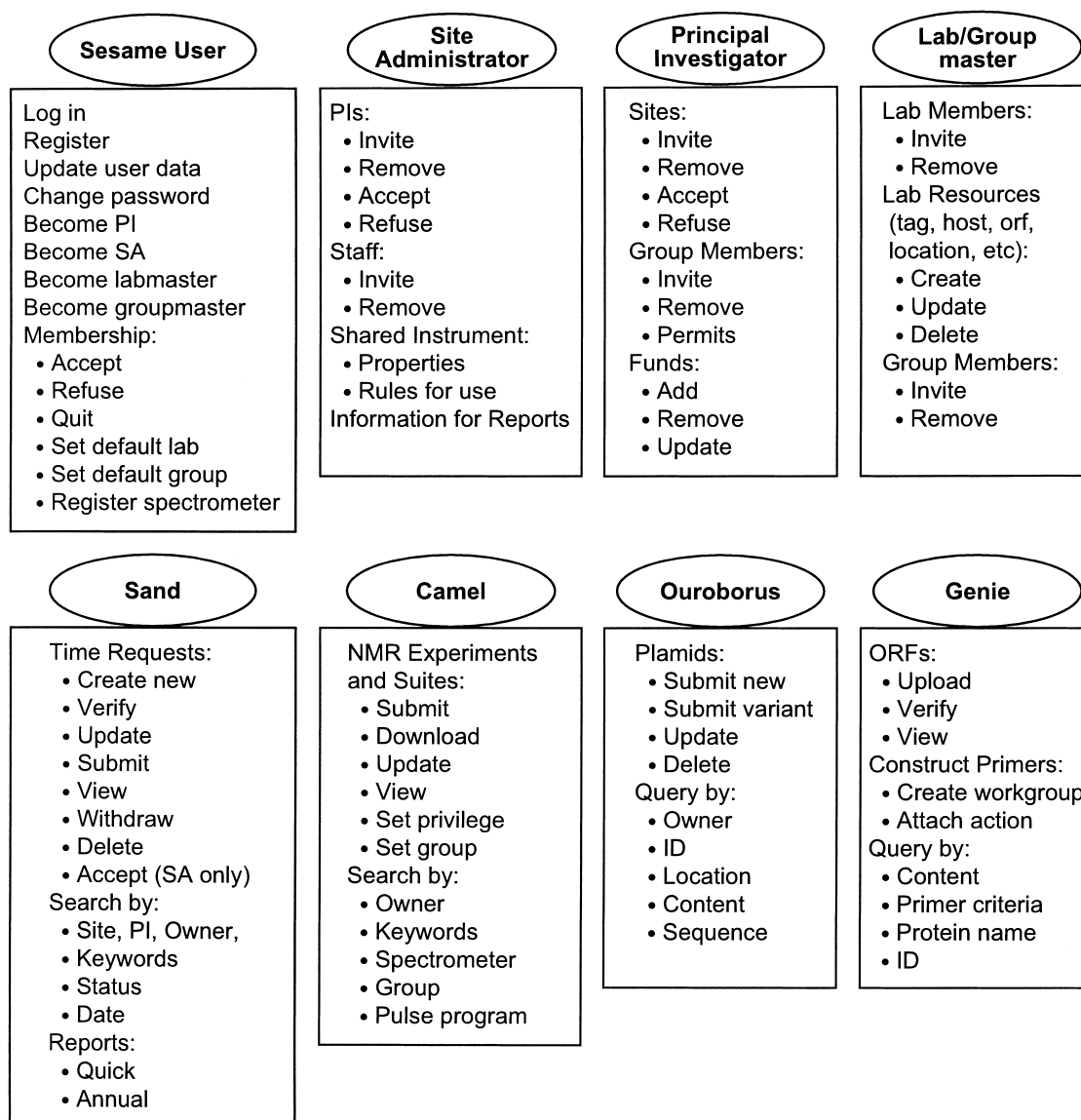


Figure 4. Examples of actions that involve interactions with the Sesame database.

Results

Sesame framework

The Sesame framework has been implemented as a three-tier Java2/CORBA client/server application. Each application module has a user interface written in the platform-independent Java2 language. The application-applet for each module is accessible from the CSG homepage <<http://www.uwstructuralgenomics.org>>. The client runs on any computer with Java Web

Start, or a recent copy of a web browser that supports the Java2 plug-in. The client and server objects connect to one another across the network via Object Request Brokers (ORBs). The client's interface to the ORB is a stub that is written in Interface Definition Language (IDL), while the ORB's interface to the server is through an IDL skeleton. The IDL provides a programming-language-independent mechanism for describing the methods of an object, while the skeleton provides programming-language-independent access. The application server maintains the connec-

tion to the database, assembles and executes the SQL statements, and works with the result sets. Database programming uses the Java2 JDBC API, which decouples the database management system (DBMS) from the client and makes changes in the DBMS invisible to the user. The current database server is Oracle 8.1.7 Enterprise Server; however, the system can be easily adapted to other DBMS software. The Sesame schema is a partially normalized relational database model tuned for maximum performance.

Administrative functions

The administrative tools in Sesame have been designed to handle a wide range of situations. A typical structural proteomics project will make use of only a subset of these. Additional administrative tools allow Sesame to support multiple independent structural biology projects in ways that provide them access to technology developed for the CESG. To use the Sesame experiment management system, one has to be registered. Users register themselves during their first access to the system by supplying the following basic information: user name, password, full name, e-mail, and organization. The password is encoded in the client, using the MD5 message-digesting algorithm, and the unencoded form never leaves the client. Only the encoded password is stored in the user's profile in the database. In analogy to operating systems such as Unix, Sesame has multiple privilege levels (user, laboratory, group, world, principal investigator, site administrator, etc.).

Objects in a Sesame laboratory (lab) can be thought of as those available to all lab members (reagents on shelves, laboratory instruments, shared laboratory protocols, etc.). Objects attributed to a given lab are visible to all lab members but only to them. The lab resources (lists of stock solutions, restriction enzymes, experimental protocols, etc.) are maintained by the 'lab master', someone designated to organize the laboratory. At the moment, lab data cannot be made visible to the world (future versions of the software will include an option to make some lab-related data visible to the world). In a lab, each record belongs to the person who initiated it. Other lab members can copy it (e.g., primer or plasmid entry, order form), use it as is under their ownership (e.g., carry out the same protocol on their protein, or reorder the same widget for themselves) or use it as a

template, modify it, and save it for their own use. Only the owner can update or delete a record.

Sesame modules can be configured to provide security at different levels: individual, defined group, or unsecured ('World'). All users are assigned by default to the 'World group' and 'World lab'; the parts of Sesame that are accessible to anyone connected to the Internet. Any person can create a 'group' and define a 'lab' within Sesame. Persons are added to a group/lab through invitations issued by the founder of the group (or designated administrator). Invitees become members of the group/lab only upon accepting the invitation. Information that is to be freely disseminated is placed in the World group.

The privilege level associated with stored information regulates its access. Files belonging to a particular group/lab are visible to members of that group/lab but not other users. Files belonging to an individual alone are visible only to that individual (its owner). Users have no indication of the presence of information outside of their privilege.

Only the owner can modify the information in a file. However, users who have access to particular information can use it as a template, upload it, and become the owner of a copy stored in another location. The user can store the copy as-is or modify it prior to storage. An owner can relax the current restriction on a file, by setting the 'group' or 'public' flag. If a record/file is public, it is viewable/downloadable by anyone, including a newly-registered user. The group flag helps to share experiments among virtual (e.g. far-flung collaborators on projects) or geographical (e.g. facility, department) groups. Group members can resign from a group at any time. Also the organizer (owner) of a group can remove any of the members. This way, information sharing can be controlled fully at any level. The database can be searched on the basis of different fields, such as 'owner name', 'experiment name', 'equipment type', 'protocol', 'keyword', 'protein name', etc. The results of each query are privilege filtered; i.e. if the person who issued the query has no privilege to view a record, it will not appear in the query result list.

Application modules

Released versions for nine (web-accessible) Sesame application modules are currently in use: Genie, Ouroborus, Lamp, Jar, Camel, Sheherazade, Sand,

Sundial, and Well. All are publicly available from the Sesame web site <<http://www.uwstructuralgenomics.org>>. An additional module described here (Bazaar) has been designed and is under construction. The Sesame help pages (accessible from the Sesame web site) show representative screens and illustrate how users can interact with individual modules.

Genie is the Sesame module dedicated to tracking the progress of protein production projects. Genie is organized by genome and contains available sequence information and annotation captured from genomics and bioinformatics CESG collaborators plus information on open reading frames (ORFs) added by the CESG staff. Information on each ORF includes: number of predicted introns, genetic code and number of bases, amino acid sequence and number of residues, SwissProt identifier, cluster of orthologous groups (COG) identifier, source organism, chromosome, gene locus, gene locator, protein category, predicted class, protein name, pointer to a structure of the protein (or homologue) if deposited in a public data bank, indication of whether the protein (or homologue) is under study at another SG site, and the scores that CESG uses in ranking targets. ORF records can be searched by IDs, name, number of introns, sequence length, criteria for primer construction, etc. The user can select ORFs for experimentation by assembling them into “work groups” of a user-defined size (e.g. for a 96-well plate). The user can construct primers (catalogued and viewable with the Ouroboros module), or apply a series of actions to the work group and create a computerized record of the cloning, expression, and protein purification for the work group.

Ouroboros is the module designed to manage primers and expression plasmids made for producing proteins. The expression plasmid records contain the following data: protein name, variant name, host, plasmid name, plasmid source, construct, database number, parent number (to permit the tracking of variants), location (room, freezer, tower, etc.), genetic code and amino acid sequence (determined from the genetic code), and details about the construct. Figures can be appended to the record, for example a diagrammatic illustration of the construct. The data in the primer records are: primer name, user label, identifier from the laboratory synthesizing the DNA, date synthesized, purpose, genetic code, location of the product, and additional comments. The primer records can be linked with the plasmid records, allowing users to easily look up what primers were

used for a given plasmid, or what plasmids were made using the given primer. Lab members can search the database according to fields such as ownership, location, construct, database number, parent number, amino acid sequence, etc. The lab master maintains the lab resources, such as lists of plasmids, hosts, restriction enzymes, tags, and cleavages used in the lab. All these lists contain the names of the given items, additional information, amino acid or DNA sequence, and the URL of the vendor. A report can be generated about the expression plasmids and primers.

Lamp is the module that manages information about the expression and purification of proteins and organizes their storage. Lamp records contain data about protein name, original source, user label, link to the expression plasmid, link to the NMR samples made from it, tree type protocol description (with user definable nodes and comments attachable to every node), and gel scan and chromatography figures. The lab members can search these records by various criteria (location, content, plasmid source, etc.). The lab master maintains a new lab resource accessible by lab members only: recipes (e.g., compositions of particular buffers, standard procedures used in isolating and purifying proteins). A report file can be generated for every entry. Lamp can also show the data related to the currently selected protein from Ouroboros (plasmid used to make the protein) and Jar (samples made from it).

Well. The Well module allows users to design screens for crystallization trials and to optimize the crystallization conditions. It keeps a detailed description of each protein sample and the conditions of the crystallization trial (protein from Lamp, type of derivative or labeling, pH, salt, date, temperature, additives, screen type, size and volume, location of the bar-coded plate), along with information on the observed results and progress toward obtaining diffraction quality crystals. Also, the module is able to create an input (control/parameter) file for the CyberLab (Gilson, Middleton, WI) crystallization robot.

Jar is the module designed to manage NMR samples. Jar records contain data about protein name, user label, sample type, description of the components (name, concentration, isotopic labeling), sample conditions (pH and ionic strength), lock solvent, additional comments, link to protein record, and links to the NMR experiments for which the sample was used. The lab members can search these records by various criteria (location, content, ownership) and search for

all NMR experiments performed on a given sample. The software is capable of generating a complete report on each sample.

Camel is a program that manages NMR experiments. It provides a solution to the problem of storing, recovering and disseminating (laboratory-wide or worldwide) parameters used in NMR data acquisitions. Camel has the ability to store parameter sets into a relational database, save sets to a similar spectrometer, and set up collaborative groups for sharing sets. With Camel, the user can access the centralized database, query it, and restore files and/or directories needed to rerun the selected experiment(s). The downloaded environment can also serve as a starting point to set up a new experiment. Camel can also link to a particular NMR sample (described by Jar), and attach NmrPipe scripts and Felix macros (to process the raw NMR data) to a given experiment. The Camel module is also an example of how a separate script-derived software package can be interfaced to the experiment management system. Camel provides representations of the parameters utilized by the Bruker or Varian software package in launching and carrying out a data acquisition protocol. Camel has been in use at NMRFAM since May 1999 and more recently at the Medical College of Wisconsin where it provides the mechanism for capturing every parameter associated with NMR data collection and dissemination of these data to all researchers involved in the structural proteomics project at multiple institutions.

Sheherazade is the master client module. Currently Sheherazade integrates Ouroboros, Lamp, and Jar) and makes it possible to traverse steps in a proteomics project, both horizontally (following all the links and branches between different steps) and vertically (viewing and editing all the available records in a particular step of the project).

Sand is the application module used for requesting and allocating data acquisition time on NMR spectrometers. Sand has been in use by all users of the National Magnetic Resonance Facility at Madison since October 1999 and more recently at the Analytical NMR Facility, Mayo Clinic and Foundation, Rochester, Minnesota. Sand provides multiple levels of access and functionality to users, principal investigators and site administrators, while providing complete data security throughout. The incorporation of a relational database server allows for sophisticated querying and the ability to generate summative reports at a keystroke, not only for NMRFAM staff members, but also for principal investigators and pro-

gram administrators. These include the ability to gather statistics by a principal investigator of his/her group's usage of the Facility. Program administrators have the ability to query the database to obtain information on demographics of Facility users, titles, abstracts, and funding sources of projects, and data collection time used on various spectrometers. Since the Sand module was installed at NMRFAM, over 1900 separate, accepted, time requests have been processed. The report generation features of Sand have been used to produce the major part (typically 140–150 pages) of the NMRFAM Annual Report. These include descriptions of user projects, tables of spectrometer use, statistics on data collection at each spectrometer, and specialized information on each project required by the granting agency. This feature of Sand yearly saves several days of tedious work.

Sundial. The Sundial module is used by the Site Administrator of an NMR facility to schedule time on the facility's spectrometers (seven spectrometers in the case of NMRFAM). Sundial communicates with the Sand module to collect all requests for time for a given spectrometer over a given time period. The administrator then maps those requests onto a calendar. The finished schedule is then released to the public. Individual users are able to view their allotted time, and sign up for additional time and relinquish time slots that prove to be unnecessary. The module saves time in organizing the utilization of a scarce and expensive resource and helps minimize unused time resulting from cancellations.

Bazaar. Bazaar has been designed to capture information about orders for supplies and equipment, to track their status, and to associate charges with particular functions of a project. With Bazaar, project members will be able to access previous orders and replicate information needed to reorder from a given vendor. Administrators will use Bazaar to track expenditures and account balances and to create reports of expenses according to subproject or other category.

Discussion

Architecture and platforms

The multi-tier client/server paradigm was chosen as the basis for the Sesame project management system because it supports light clients that can be located anywhere an Internet connection is available. Within

a laboratory, connections can be on a local area network, either hard-wired or wireless. All laboratories participating in a project have the same data views and processing capabilities available to those at the home site. Our goal was to develop a robust, scalable infrastructure that would support the Sesame three-tier architecture along with the computational and storage demands of CESG, NMRFAM, and additional users. High availability (>99%) and low fault tolerance are achieved through redundant hardware, RAID systems, load balancing, and appropriate software mechanisms. The hardware is housed in a controlled environment with thermal stability and limited access.

The network server is a set of high performance computers interfaced by very high speed connections (e.g. Gigabit-Ethernet, fiber optic); this set of computers functions as application or storage servers. The application servers need to be high performance multiprocessor computers with large memory to be able to support multiple clients executing different methods of different complexity, or they can be a large array of lesser performance computers where each of them would support only a small number of clients (application dependent) per processor. The application servers can include a server cluster or a network of grid-enabled computers. The storage servers are high-performance fault-tolerant file servers with capacities in the terabyte range. In our laboratory, the storage server constitutes a common shared resource.

The client computers provide access to data and applications served by the network servers. They load (from the network server) and execute only the client (light) part of an application, while the server part is executed on the network application servers, and the data are fetched from a database and/or files that reside on the storage servers.

This architecture has a number of advantages. The state of a properly designed client application is maintained on the server. This ensures a high level of fault tolerance because the network application does not depend on the particular client computer. Server administrators maintain the network applications. Because common resources are employed, the network applications benefit from greater online collaboration and communication. Network applications can balance flexibility and ease of use with standardization. As applicable, users are free to tailor the client side of the network application to best suit their needs. This organization removes problems associated with file dissemination among users running different software on different platforms. When a new

version of a network application is installed, it is immediately available to the whole user base. Because the application is installed on network servers, users do not waste time maintaining software locally.

The architecture does require that professionals develop the infrastructure and maintain the servers. Also, a continuous and reliable network service is essential, including redundant servers and network topologies, fault tolerant storage, and appropriate routing strategies.

Comparison with alternative approaches

Sesame is more powerful and more flexible than other general approaches that we are aware of for organizing experimental environments. Commercial laboratory information management systems (LIMSs) <<http://www.limsources.com/home.html>>, have been designed primarily for use in assay labs and process development environments, rather than research environments. LIMSs are too rigid to provide useful support in a setting where the research effort involves the development or utilization of new experimental protocols, including different combinations of protocols.

The SDSC Biology Workbench <<http://workbench.sdsc.edu/>> illustrates an approach in which various knowledge databases, simulations, and graphical displays can be linked together by an interface that allows their operation from a simplified user interface. As in Sesame, the output from one package in the Biology Workbench can be used as the input for another.

The European Bioinformatics Institute is developing a Java/CORBA based application wrapper designed primarily for sequence analysis software tools. The capability of this package, called Applab <<http://industry.ebi.ac.uk/applab/>> currently is limited to remote running of command-line-driven applications.

The Pacific Northwest National Laboratory (PNNL) has developed the DOE 2000 Electronic Notebook Project (CORE2000) <<http://www.csm.ornl.gov/enote/>>. The approach differs from Sesame in that it does not use defined objects stored in a database.

The needs in the biomolecular NMR field for standardization of data exchange formats and for software tools to be used in data harvesting have been recognized by funding of the CCPN project. An interna-

tional group has been meeting to develop a data model and standards [3].

Other structural genomics pilot projects are beginning to develop their own solutions to the problems of information management. For example, the Northeast Structural Genomics Consortium has an integrated tracking database called SPINE [4]. Representatives from various structural genomics groups met for an ISGO Workshop in San Antonio, Texas, in May 2002 to discuss problems and potential solutions. Among the software solutions described at the meeting, Sesame was unique in its ability to support multiple research groups with flexible levels of secure access and in its use of ORB technologies.

Three non-ORB technologies employed by others (html, cgi, and servlets) are less powerful and flexible than the approach used in Sesame. As discussed by Orfali and Harkey [2], the advantages of CORBA over alternative approaches are that: (1) the marshalling and unmarshalling of parameters need not be provided; (2) interface descriptions can be given to provide higher levels of abstractions, and parameter typing is supported; (3) distributed method invocations and transactions can be performed; (4) dynamic discovery of a remote object's interface is supported, as are dynamic invocations; and (5) states are easily maintained across invocations and even across sessions. The invocation performance of CORBA is an order of magnitude faster than for servlets, and two orders of magnitude faster than for CGI. Sesame is a true distributed application, and the CORBA middleware ensures that active Sesame sessions are decoupled from the web server, and unaffected by the web-server's state (restart, or denial of service attack). By contrast, CGI and servlet mechanisms are an extension of a web server, and hence are tightly coupled with it. In comparison with other ORB middleware, DCOM (Distributed Component Object Model) and RMI (Remote Method Invocation), CORBA is the only one that is independent of both operating system and language and has persistent object references. An important future advantage of CORBA-based Sesame is that it will interface easily with the CORBA-based Protein Data Bank server [5] and with CORBA-based software being developed by instrumentation manufacturers.

Conclusions

The Sesame framework and modules developed to date serve as a proof of principle for the general approach. Sesame supports information tracking on genome-wide scales. It permits appropriate views of individual project for workers at the bench and for administrators. The system is easily interfaced to laboratory instrumentation and to barcode readers and printers. Sesame supports control of software packages used in data collection, processing, and analysis.

Goals for the future are to design and implement additional modules needed to complete the seamless tracking and control of all steps in a structural proteomics project, from target selection to deposition of results in public databases and publication. As the development proceeds, modules that address perceived bottlenecks will be given higher priority. The scope and standards for data output will conform to those being developed by the 'International Structural Genomics Organization (ISGO) Task Force on Deposition, Archiving, and Curation of the Primary Information.'

The Sesame schema currently consists of more than 90 tables. We plan to make the CORBA data structures and methods interfaces available publicly through the Sesame web site. All Sesame modules become available to the whole community at the moment they are deployed on the web. The advantage for users is that they always are presented with the latest and greatest version of the system. Both Java Web Start, and the Java Plug-in have the capability to cache the current version of the module locally. During start-up the software first checks to determine if a newer version is on the server. If so, the new version is downloaded; if not, the locally cached version is run. Hence, a Sesame user is not required to perform an explicit upgrade. An option allows users to create a formatted text file dump of the items owned. Future versions of Sesame will contain a tool that enables users to back-up their data locally as a tagged file for entry into a personal RDBMS. Another plan is to provide single-user, non-distributed versions of Sesame modules that will operate on a personal RDBMS.

The Sesame project management system is accessible for use through the home page of the Center for Eukaryotic Structural Genomics at the University of Wisconsin-Madison <<http://uwstructuralgenomics.org>>. Its existing modules can be used remotely. Information on software availability and instructions for installing Sesame on remote servers is available free-of-charge to academic users, although they must license third-party software packages such as the RDBMS.

The use of Sesame should facilitate collaborations and longitudinal investigations. All coworkers with a need to know can have access to a history of what has been done and all results achieved on a given project or set of projects. Scientists following up on previous work can quickly review the results along with the protocols used to achieve them. We are already finding the database to be a rich resource for evaluating hypotheses concerning strategies for high-throughput proteomics.

Acknowledgements

Sesame is an outgrowth of a project supported by the National Science Foundation (BIO/IID/IDBR grant DBI-9604906) that was based on the 'ZOO database tools' developed by Prof. Miron Livny (Computer Sciences Department, University of Wisconsin-Madi-

son) and Prof. Yannis Ioannidis (Computer Sciences Department, University of Athens). The current project received support from the National Institutes of Health (National Center for Research Resources grant P41 RR02301-S and the National Institute for General Medical Sciences grant GM P50 GM64598). We are indebted to Dr. William H. Clifton and his colleagues at the LEAD Services (College of Engineering, University of Wisconsin-Madison) for their professional evaluation of the Sesame project.

References

1. Edwards, A.E., Arrowsmith, C.H., Christendat, D., Dharamsi, A., Friesen, J.D., Greenblatt, J.F., and Vedadi, M. (2000) *Nature Struct. Biol.* **7** supplement, 970-972.
2. Orfali, R., and Harkey, D. (1998) *Client/Server Programming with JAVA and CORBA*, 2nd edn., John Wiley and Sons, New York, NY.
3. Fogh, R., Ionides, J., Ulrich, E., Boucher, W., Vranken, W., Linge, J.P., Habeck, M., Rieping, W., Bhat, T.N., Westbrook, J., Henrick, K., Gilliland, G., Berman, H., Thornton, J., Nilges, M., Markley, J., and Laue, E. (2002) *Nature Struct. Biol.* **9**, 416-418.
4. Bertone, P., Kluger, Y., Lan, N., Zheng, D., Christendat, D., Yee, A., Edwards, A. M., Arrowsmith, C. H., Montelione, G.T., and Gerstein, M. (2001) *Nucleic Acids Res.* **29**, 2884-2898.
5. Berman, H.M. Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., and Bourne, P.E. (2000) *Nucleic Acids Res.* **28**, 235-242.