Journal of Digital Imaging

MIMI: Multimodality, Multiresource, Information Integration Environment for Biomedical Core Facilities

Jacek Szymanski,[1] David L. Wilson,[2] and Guo-Qiang Zhang[1]

The rapid expansion of biomedical research has brought substantial scientific and administrative data management challenges to modern core facilities. Scientifically, a core facility must be able to manage experimental workflow and the corresponding set of large and complex scientific data. It must also disseminate experimental data to relevant researchers in a secure and expedient manner that facilitates collaboration and provides support for data interpretation and analysis. Administratively, a core facility must be able to manage the scheduling of its equipment and to maintain a flexible and effective billing system to track material, resource, and personnel costs and charge for services to sustain its operation. It must also have the ability to regularly monitor the usage and performance of its equipment and to provide summary statistics on resources spent on different categories of research. To address these informatics challenges, we introduce a comprehensive system called MIMI (multimodality, multiresource, information integration environment) that integrates the administrative and scientific support of a core facility into a single web-based environment. We report the design, development, and deployment experience of a baseline MIMI system at an imaging core facility and discuss the general applicability of such a system in other types of core facilities. These initial results suggest that MIMI will be a unique, cost-effective approach to addressing the informatics infrastructure needs of core facilities and similar research laboratories.

KEY WORDS: Computer system, data collection, data extraction, image data, image distribution, information management, information storage and retrieval, information system, internet, management information systems, open source, cost analysis, biomedical core facilities, unified modeling language

BACKGROUND

Modern biomedical research is inherently multileveled and multidisciplinary. To facilitate this research, core facilities bring the latest imaging and scanning technologies to the research community and support many projects simultaneously. However, they often do so in the midst of significant information management challenges unforeseen at their inception, such as:

- Effective and efficient distribution of acquired scientific data from a core facility to its investigators
- Timely sharing of raw, primary, and curated data for collaborative activities
- Optimized scheduling and resource usage
- Management of experimental workflow, e.g., multiple related steps in one-time or longitudinal studies
- Management of administrative workflow, such as tracking of material cost, staff times spent on sample preparation and data acquisition, and billing and accounting
- Monitoring of the overall resource usage of a core facility, by compiling, e.g., a profile of usage statistics of equipment and types of involved projects

[1]From the Department of Electrical Engineering and Computer Science, Case Western Reserve University, 10900 Euclid Ave, Cleveland, OH 44106, USA.

[2]From the Department of Biomedical Engineering, Case Western Reserve University, 10900 Euclid Ave, Cleveland, OH 44106, USA.

Correspondence to: Jacek Szymanski, Department of Electrical Engineering and Computer Science, Case Western Reserve University, 10900 Euclid Ave, Cleveland, OH 44106, USA; tel: +1-412-6149501; e-mail: jxs186@case.edu

Online publication 13 November 2007

doi: 10.1007/s10278-007-9083-y

- Coherent and common access point for data analysis workflow, linking raw data and/or primary data with results from analyses, reports, images, and references, and comparing with related results from existing databases and literature

There are currently no comprehensive software systems addressing these challenges as a whole (Siemens' MIPortal[1] focuses on improving the management of experimental workflow for proteomics research and does not address administrative issues). From our prior experiences with the Case Center for Imaging Research (CCIR), as well as from information gathered from many core facilities around the country, deficiencies with the existing infrastructure are often manifested in:

- Substantial administrative and personnel overhead. This exists in pen-and-paper-based record keeping aided by disconnected spreadsheet programs, manual management of scheduling on a common off-the-shelf calendar system that operates in isolation, using portable media for data transport, and relying on e-mail communication to gather a variety of project-related information. Some centers operate under an information technology (IT) infrastructure resulting from adopting/adapting existing open-source/inhouse/commercial software for managing a variety of data, although this only reduces the problem to the equally, if not more, challenging issues of information integration, interoperability, and resource for IT personnel support.
- Lack of support for collaboration among researchers. The disintegration of administrative and scientific data makes it difficult to access data and find information about related prior studies. Collaborating researchers must then rely on ad hoc mechanisms such as e-mail communication to share data and results. This not only makes the bookkeeping of data a chore, but it also lacks a uniformly enforceable standard for the safety of valuable data and results from analyses.
- Significant amount of redundant, disintegrated, and inconsistent data. When data are kept in disconnected systems, information such as a principal investigator's profile and projects may have to be reentered multiple times to multiple systems, making it difficult to maintain and update. Repetition in data entry not only requires additional effort, but it also opens more room for errors and inconsistencies: the same entities may have been entered using different names in different systems, and changes made in one system may not automatically propagate to other systems.
- Lack of support for the integration of information from disparate resources. Access to data and knowledge is often labor-intensive, repetitive, disorganized, and burdensome; project management and data analyses are tasks relegated to individual investigators without a common framework or standard for record keeping or for sharing and collaboration using intermediate results.

The root cause for these deficiencies can be summarized as a lack of a holistic approach to infrastructure support. Given the challenges encountered by imaging and other kinds of core facilities, an approach that captures a vision for a long-term solution and addresses some of the immediate needs is desirable. This paper describes a system called MIMI (multimodality multiresource information integration environment) that not only addresses some of the needs and provides a flexible and expandable solution to the challenges mentioned above, but also provides a foundation for a more advanced system that substantially integrates existing knowledge with analyses and curation of experimental data. We report the design, development, and deployment experiences of a baseline Imaging MIMI system at the CCIR, a typical large-scale imaging core facility. The main features of the Imaging MIMI system are:

- Effective, efficient and secure data storage and archiving of a variety of imaging data (e.g., digital imaging and communication in medicine)
- Web-based access of acquired imaging data by researchers unconstrained by time and location
- Sharing of raw and primary imaging data among collaborators
- Resource scheduling and management
- Monitoring of the overall resource usage of a core facility, by compiling, e.g., a profile of usage statistics of equipment and types of supported projects
- Built-in mechanism for tracking regulatory and operational qualifications [e.g., Institutional Animal Care and Use Committee (IACUC)]

The Imaging MIMI system comes with a web-based interface to support core membership and project information management. It features an expandable and modifiable framework that can adapt to the needs of imaging and other kinds of core facilities. In Methods, we describe our developmental guidelines and MIMI's system design and implementation. In Results, we present usage statistics automatically gathered by MIMI's built-in tracking tool. We also provide a cost–benefit analysis to demonstrate the realized and to-be-realized tangible benefits. In Discussion, we give concluding remarks.

METHODS

Guiding Principles and Approach

In developing MIMI and its particular realization for imaging research, we have adhered to the following set of guiding principles:

- Use an open-source environment for development
- Fully integrate the biomedical enduser into the developmental team
- Maintain uniformly web-based, menu-driven, friendly user interface
- Decentralize data and information management tasks with role-based access control, semiautomated data flow, and resource scheduling to minimize overhead after deployment
- Employ the latest methodologies and tools in IT and software engineering in software development

The choice of an appropriate open-source developmental environment[2] not only saves developmental cost, but also ensures that the system is modifiable and expandable without proprietary restrictions. The potential downside of a steeper learning curve and the stability of the supporting community can be overcome by a careful scrutiny of the available open-source packages and suitable training of the programmers. MIMI uses Plone,[3] which is an open-source content management system, as its main developmental environment. We chose Plone for its web-based interface for development and its built-in web-server incorporating the latest techniques for content-management, such as version control and cascading style sheets (CSS).[4] Plone's object-oriented framework allows rapid development through code reuse and extension of proven functional modules. The object-oriented paradigm allows objects placed insider other objects (such as folders) to inherit and reuse their parents' attributes, contents, and functions. Plone's object-oriented framework extends to the storage level, allowing developers to conceptually organize information in a logical manner that in turn speeds-up development. The Plone distribution is available for major operating systems such as Mac OS, Windows, and Linux, so a developer can select a preferred environment for development. Our earlier successful experience with Plone[5] also made it an easy choice.

Fully integrating the enduser into the developmental team ensures usability, relevance, and impact to the targeted application domain. Although neither consciously nor strictly following the extreme programming[6] practice, we find it extremely important to engage the enduser into all steps in the software development process. The engagement of the enduser helps realize two of the core values of extreme programming immediately: communication and feedback. Through regular meetings, ongoing changes to loosely specified requirements occur as a natural process. The adaptability to changing requirements is a more realistic and better approach than attempting to define all requirements at the beginning of a project, because the developer and the enduser rarely have complete foresight of the desired end product at its inception. Rather, the ongoing discussions become a cooperative activity that helps define, refine, and deepen the understanding of what is desired. However, discussions alone without a concrete system would not be effective. This brings us to the second aspect related to extreme programming: test-driven development. Although the goal of test-driven development is to make sure that current code meets requirements, we use these informal tests as a way to demonstrate the features and functionalities of the system to generate in-depth, timely, and specific feedback to the developer. Of course, any unusual behavior of the system will show as bugs or defects to be corrected for the next iteration of demonstration. Depending on the workload and available manpower, these live demos of partial working systems can happen on a weekly or monthly basis. Not all projects can enjoy the luxury of close participation with the enduser, and we are in a unique position to have the commitment from the researchers and staff members of the CCIR to participate in developing a comprehensive set of features in a baseline Imaging MIMI system that cannot be found in a single existing system.

The remaining three principles of web-interface, decentralized content management, and employing the latest technology are common-sense: the web-interface provides uniform and wide accessibility; menu-driven interaction provides more control over data input, output, and presentation; and decentralized content management reduces the overall management overhead after the system is deployed. However, achieving these requires a long-term vision and knowledge in several related fields. We are again in a unique position to follow these principles during development due to Case Western Reserve University's distinct collaborative environment across management centers. Our guiding principles should be relevant and helpful for similar efforts.

System Overview

The baseline MIMI consists of two main components: the Meta Server and the Data Server. The Meta Server is the common front-end for MIMI's functionality. It is called "Meta Server" due to its role in managing all relevant alphanumeric data: user profiles, project information, scheduling information, data storage address information, access control, etc. It supports a web interface for data downloading after experimental data is acquired, using the client–server paradigm.[7] Administrative functionalities are also support by the Meta Server, such as validating user-supported information, assigning access privileges, and confirming requested scanning sessions. In a manager's role, a user can launch the usage-statistics program to monitor resource usage and generate statements for fees for the core. The Meta Server is also involved in the final step of data flow: after imaging data are acquired, a Java program can be launched from the scanner work station (usually a PC), which receives input about the address of a local folder containing the acquired data and a redundant array of independent disks (RAID) directory path on the Data Server representing the location where the data will be stored. The RAID path consists of metadata automatically generated by the Meta Server to represent the unique, humanly readable directory path on the Data Server.

The Data Server is the backend for storage management of acquired image data. It uses a standard folder hierarchy for storage. To safeguard data from network viruses[8] and prevent unauthorized access, the Data Server operates behind a hardware firewall with communication permitted only with the Meta Server and with the local area network (LAN) PCs attached to scanners. The Meta Server and the Data Server together achieve common functionalities of a data warehouse.[9]

Design and Implementation

MIMI is designed to support a core facility's administrative and scientific workflows in a single system. The administrative workflow includes managing profile data on users and research projects, scheduling scanning sessions, billing services, and compiling performance statistics to monitor resource usage. The scientific workflow consists of managing scientific data and disseminating them to the relevant researchers through a common web-interface.

Three data models will be used for the administrative workflow (Figs. 1, 3, and 4). The description of these data models follows the activity diagram specification of the Unified Modeling Language (UML).[10] A solid dot represents the initial state. Rectangular boxes and round-corner boxes denote activities and objects, respectively. Solid arrows specify transitions between activities. Dashed arrows represent interactions with objects, ie, dashed arrows entering or leaving an object represent modification/creation or retrieval, respectively.

We begin by explaining how each data model supports the administrative workflow. We then describe how each data model is implemented using Plone. We conclude this section by showing how the scientific workflow is addressed through the data-flow model (Fig. 5).

User Profile Model

The user profile model (Fig. 1) specifies the behavior of the user management segment of the administrative workflow. The model is aimed to ease the data entry burden of a core facility by allowing the user to enter data, which will be

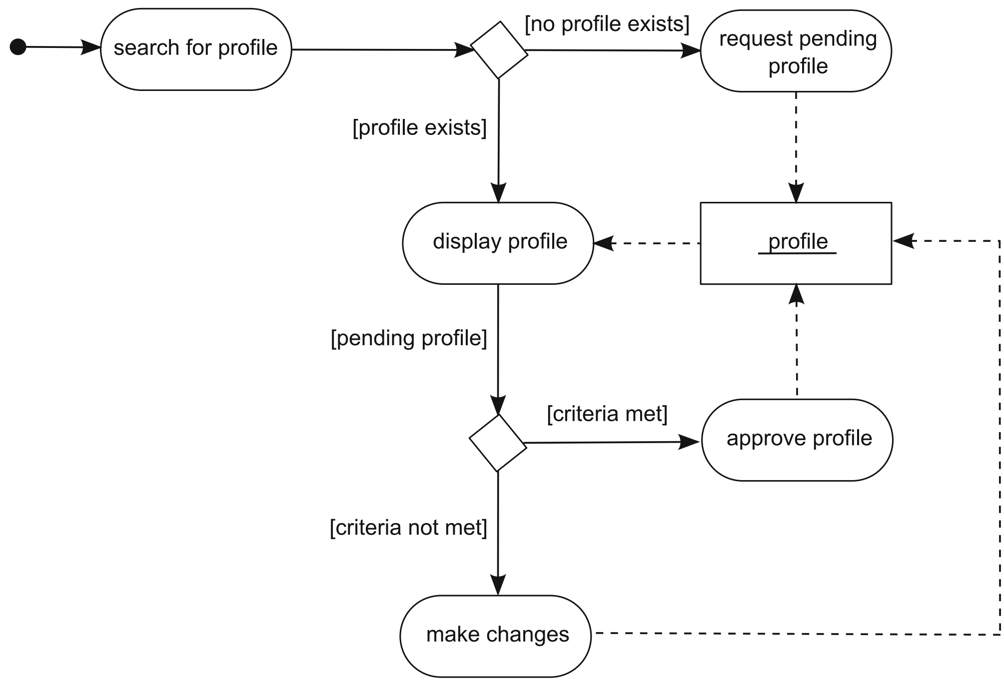

Fig 1. The user profile model.

validated by a manager in a core facility for it to become effective. The first action indicated by the model is searching for an existing user profile for a specific user. If the user profile does not exist, then it must be requested by the user as a pending user profile. Otherwise, if the user profile exists, then it will be displayed. The user profile model then proceeds to define actions for a pending user profile. A pending user profile that does not meet the criteria for approval needs to be modified by the user or a core facility manager. A pending user profile that meets the criteria for approval can be approved by a core facility manager.

The Plone implementation of the profile model uses the profile object, which stores details about core facility users. The profile object resides at the top level of the Plone object hierarchy (Fig. 2). It captures information using the following string attributes: first name, last name, e-mail address, institution, department, phone, fax, address, city, state, zip code, country, login ID, and status. The last two attributes store a user's ID for logging into Plone and a value for pending (P) or approved (A) status, respectively. The profile object also contains a roles attribute that stores a list of user roles. Plone accesses the value of the roles attribute to determine a user's access privileges. The four possible user roles are Principal Investigator (PI), Coinvestigator (CI), Operator, and Manager (a user can assume multiple roles). Users with the PI role are researchers who have active research projects. Users with the CI role are collaborators who work with other researchers. Operators represent users who are qualified to operate equipment. Managers are core facility staff members with "superuser"

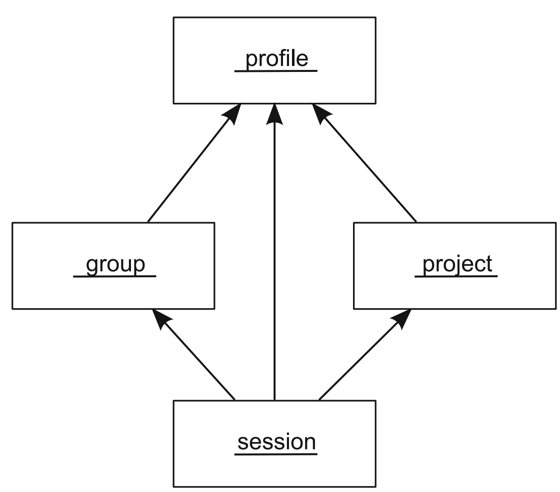

Fig 2. UML object diagram of Plone's objects and their inheritance relationships.

privileges, ie, they have access to all of MIMI's functionalities.

When a user is granted the privilege to create a profile object, a profile request form is presented with input fields to capture information such as a user's e-mail address and phone number. Once a user submits the profile request form, Plone creates a pending profile object with its status attribute set to "P." Core facility staff members with profile objects that contain "manager" as a value for the roles attribute are ultimately responsible for approving all pending profile objects. The main criterion for approval is verifying that a profile object's login ID is associated with the right contact information such as a user's e-mail address and phone number. Approving profile objects through Plone guards against malicious users who attempt to pose as others to gain access to private information.

Project Model

The project model (Fig. 3) specifies the behavior of the project information management segment of the administrative workflow. The initial state of the model consists of a decision node that returns "Yes" or "No" depending on whether an existing project is selected. If "Yes", then the information about the selected project is displayed. If "No", then a user can request a (pending) new project. The project model then specifies actions for both pending and approved projects. A pending project that does not meet the criteria for approval must be modified by the user or a manager, whereas a pending project that meets the criteria for approval can be approved by a manager. An approved project can be modified by its owner to grant privileges for specified collaborators (among the existing users) to access the associated experimental data.

The project model is implemented in Plone using a project object that captures information about a specific research project. The project object uses the following attributes to capture the associated information: name, PI, CIs, IACUC number, grant number, account number, and description. The name attribute stores the title of an active grant or a pilot study. The PI attribute, which stores the ID of a profile object that contains "PI" as a value for its roles attribute, links a project object with a principal investigator. The CIs attribute specifies a project's collaborating users

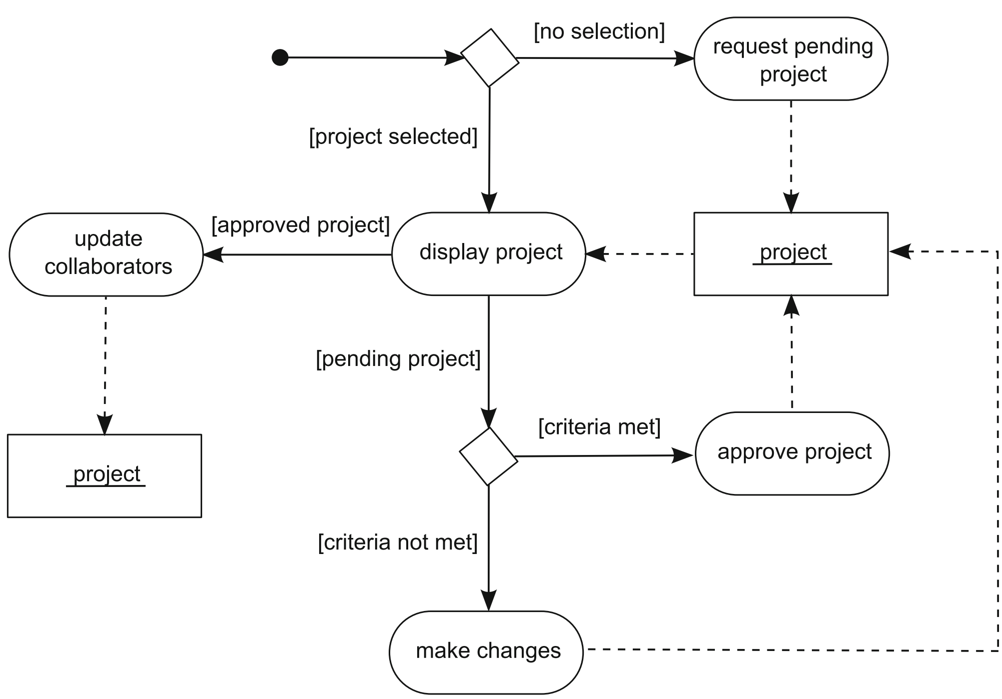

Fig 3. The project model.

by storing the IDs of existing profile objects that contain "CI" as a value for their roles attribute.

A project request form is implemented in Plone to allow a user to request a project object which, if approved, will have the user as its owner. This form contains input fields to capture project details as mentioned in the previous paragraph. It also presents a checkbox interface to allow a user to select user profile objects for inclusion as values for CIs. Once a user submits a project request form, Plone creates a pending project object with its status attribute set to "P." Core facility staff members with profile objects that contain "manager" as a value for the roles attribute are ultimately responsible for approving all pending project objects. The criteria for approval include checking that a project object's grant number and account number are valid.

A user can also use Plone to view approved project objects that he/she is associated with, ie, he/she is the project PI or a collaborator. In the case that new collaborators arrive, or old ones depart, a user may modify the list of collaborating users for these approved project objects through a web-based checkbox interface. For security purposes, a user cannot create new users (ie, profile objects) and may only select collaborators for a project from existing profile objects. Relegating the management of project collaborators to project owners (ie, PIs) is an example of decentralized content management, which alleviates the data management burden of a core facility. A collaborator of a project is typically granted the privilege to access experimental data resulting from the project.

Session Model

The session model (Fig. 4) specifies the behavior of the scheduling, billing, and usage-statistics compilation segments of the administrative workflow. The initial state of the model is a decision node to determine which actions to perform depending on whether completed or scheduled

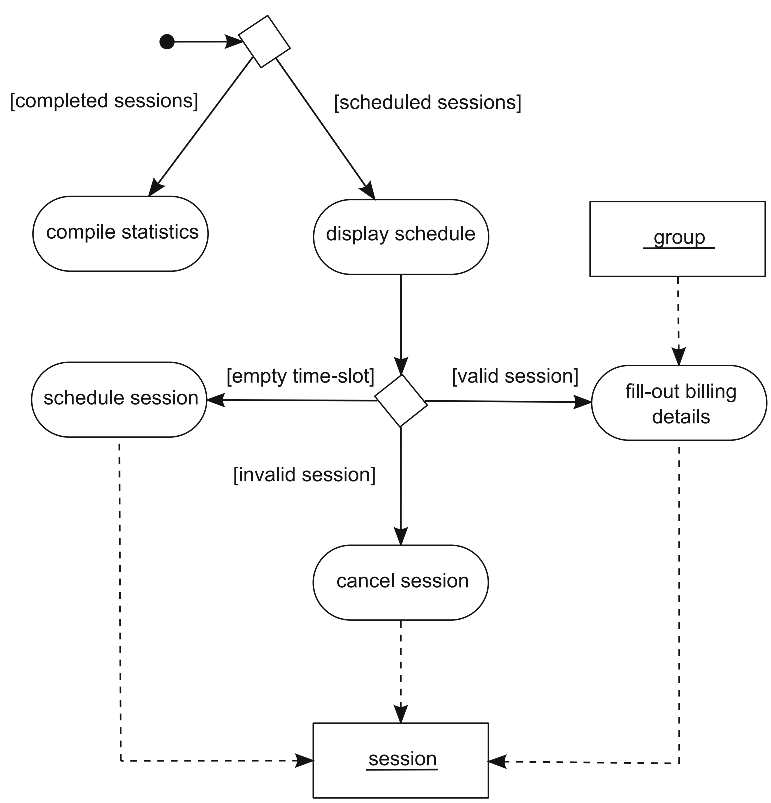

Fig 4. The session model.

sessions are selected. If completed sessions are selected, usage-statistics compilation can be performed. If scheduled sessions are selected, a calendar with the scheduling information will be displayed. The session model specifies further actions for scheduled sessions and empty time slots on the schedule. A scheduled session that is invalid (incorrectly scheduled) will be canceled, whereas a scheduled session that is valid (correctly scheduled) will be followed by input for billing details. An empty time-slot on the schedule permits the scheduling of a new session within the selected time interval.

The Plone implementation of the session model uses the session object, which represents a scheduled or completed session for an imaging system. The session object resides at the lowest level of the Plone-object hierarchy. It stores information using the following attributes: imaging system name, date, time-slot, project, operator, scanned items, time duration, total cost, status, and Data Server Path (a.k.a. the RAID path). The project attribute stores the ID of a project object related to the session object. From the project object, relevant project information such as a PI's name will be automatically retrieved and displayed on the session schedule. The operator attribute stores the ID of a profile object that has "operator" as a value for its roles attribute (the operator attribute tracks the user who operates an imaging system during a session). The scanned items attribute stores the IDs of entities (ie, small animals, cell plates) that are used during a session and the status attribute stores a value of scheduled (S) or completed (C). We will describe the Data Server Path Attribute in Data-flow.

MIMI features a web-based scheduling interface for imaging systems. The scheduling interface uses a combination of DHTML and AJAX to approximate the response speed and the look and feel of a desktop application. Users can create a new session object by dragging the mouse cursor over an open time-slot that spans at least one 30-min interval and then selecting a research project object. A new session contains values for the following attributes: imaging system name, date, time-slot, project, and status (S). A user can then use MIMI's scheduling interface to perform cancellations or to access the supplemental billing form to choose values for the remaining attributes.

MIMI's supplemental billing form contains fields that capture billing details such as time duration and cost information. It also allows a user to select the profile object for inclusion into the operator attribute. A user can also use the supplemental billing form to select scanned items by choosing group objects. A group object represents a collection of entities that have similar characteristics. For example, 20 female mice with the same vendor and strain can translate into a single group object. A group object uses the following key attributes to store information: name, species, strain, vendor, and item IDs. The item IDs attribute stores a list of unique IDs for each item of a group object. A user who submits the supplemental billing form initiates the process of automatic cost computation. MIMI then sets a session object's status attribute to completed (C) and updates the values of the remaining attributes.

The resource usage compilation capability can allow a core facility to regularly track the usage of its equipment and provide important and useful summary statistics on different aspects of the daily operations of a core. MIMI can generate performance assessments of imaging systems with different time intervals. When compiling a performance assessment, Plone locates the relevant completed session objects and sums the values of their time duration attributes. Plone's built-in search interface is modified to filter completed session objects using criteria such as principal investigator, project, and date range through text-fields and drop-down lists.

Data-flow

MIMI addresses a core facility's scientific workflow with the data-flow process. Figure 5 is a summary of the data-flow process, which is explained in more detail in the remainder of the section.

MIMI implements a data-flow process that seamlessly links imaging data with the associated session metadata. With the completion of an imaging session, imaging data are stored in a standard folder hierarchy on the attached local work station PC. The operator then double-clicks a jar file on the work station PC. The jar file is a Java executable program for the Uploader application, which is responsible for transferring the scanned imaging data to appropriate folders on the Data Server. After launching the program, the user looks up the correct session object from the Meta Server and retrieves the value of its Data Server

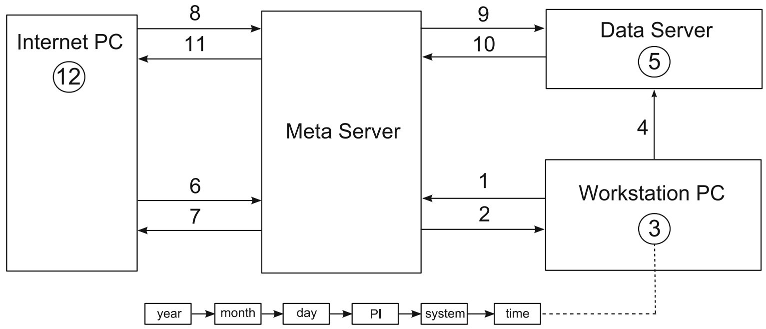

Fig 5. (1) User requests to view scheduled session. (2) Meta Server replies with session object. (3) User copies Data Server Path string into Uploader application. (4) Uploader application sends scientific data to Data Server. (5) Data Server stores scientific data. (6) Person requests to view downloading interface. (7) Meta Server replies with downloading interface. (8) User issues download request. (9) Meta Server forwards download request to Data Server. (10) Data Server sends scientific data to Meta Server. (11) Meta Server forwards scientific data to Internet PC. (12) User stores scientific data on Internet PC.

Path Attribute—a value automatically generated when a session object is created from the scheduling interface. The Data Server Path value is a string with six main parts (year, month, day, PI name, imaging system name, and time-slot) that uses the backslash as a delimiter. Because MIMI automatically accounts for scheduling conflicts, the Data Server Path value represents a unique storage location on the Data Server. The operator copies the Data Server Path value, pastes it into a textbox of the Uploader application, and selects the local directory path for the folder containing the imaging data. Once the origin and destination for the imaging data are given, the Uploader application initiates a data transfer session with a single mouse-click.

At the receiving end of the data transfer process, the Data Server runs a Receiver Script that listens continuously for requests from active Uploader applications. For all incoming requests, the Receiver Script first obtains the Data Server Path string. The script then fetches an incoming file's path and its name and concatenates them to the Data Server Path string to form an absolute storage path. The Receiver Script parses the absolute storage path into a valid folder hierarchy and creates any missing folders to form a unique storage location. The script then creates an empty file object and retrieves its contents by streaming binary data in 65,535-byte increments (the entire cycle repeats until all files transfer successfully to the Data Server).

Once imaging data are moved to the Data Server, they can be immediately downloaded by their owners and collaborators through the Meta Server. MIMI supports this step with a Retrieval Script that runs on the Data Server and listens continuously for requests by the Data Request Script that runs on the Meta Server. The communication process begins when the Data Request Script accesses a session object, obtains its value of the Data Server Path Attribute, and sends this value along with a relative folder path to the Retrieval Script. The Retrieval Script joins the Data Server Path value and the relative folder path to form a query path. The Retrieval Script then opens the query path on the Data Server and obtains a list of its files and folders (if any). The script then iterates through the list, computes file and folder sizes, and forwards these details to the Meta Server. The Meta Server dynamically constructs the visual downloading interface and sends it to the user. After the user selects files or folders to download, the Data Request Script builds a list that holds their path strings and sends it to the Retrieval Script. The Retrieval Script creates a temporary zip file and populates it by iterating through folder paths in the list and fetching any files. In the end, the Retrieval Script sends the zip file to the Data Request Script, and the Data Request Script forwards it to user's local desktop. In the case that a file is larger than 1 GB, the Retrieval Script virtually partitions the file and allows the user to download individual pieces. When a user encounters a folder larger than 1 GB, it is also possible to download only a subset of its contents at one time.

An innovative feature of MIMI's implementation of the data-flow process is the Data Server Path Attribute, which enables the treatment of imaging data as binary files. This unleashes MIMI from the complexity and variety of image file formats (ie, dcm, nifti, analyze, etc.) and avoids conversion to any standard data formats. The necessary metadata, usually stored as header information, resides in the portable path names for the folder hierarchy. It also serves as imaging data's unique IDs.

The baseline MIMI has been deployed and is in full use by the CCIR. The CCIR is a state-of-the-art imaging core facility of Case Western Reserve University's Medical Center and of the University Hospitals of Cleveland. It hosts multiple major imaging systems such as a Siemens MicroPET and a Bruker Biospec 9.4-T MR.

The Meta Server and the Data Server are deployed with a carefully chosen set of hardware and software components. The Meta Server runs on a Dell PowerEdge with dual 3-GHz Intel Xeon processors, 4 GB of DDR2 RAM, and two 300-GB 10-K RPM Ultra-SCSI hard drives. It operates using Redhat Linux and runs an Apache front-end for secure sockets layer (SSL) transmission.

The Data Server operates under the Windows 2003 operating system and provides a RAID with eight 300-GB hard drives connected with Dynamic Network Factory's 8-channel controller handling the RAID-5 functionality.

RESULTS

In this section, we present MIMI's usage analysis followed by a cost–benefit analysis. The usage analysis gives a profile of MIMI's usage statistics over an 18-month period with respect to the number of users, imaging sessions, and scientific data uploads/downloads. The cost–benefit analysis demonstrates MIMI's benefits in comparison to a status quo.

Usage Analysis

Using MIMI's usage-statistics compilation capability, we performed a usage analysis of MIMI at the CCIR. Since its initial deployment in early 2006, MIMI has served approximately 150 principal investigators, collaborating investigators, and research assistants. During this period, a total of approximately 1,600 distinct sessions have been scheduled through MIMI, spanning an 18-month period or 400 working days. This translates to four scheduled sessions per working day. Among all sessions, half are linked to scientific data. This entails that imaging data have been transferred to the Data Server using MIMI's data-flow process at the frequency of two times per working day. Users also typically download the acquired data on the same day, so data downloading through the Meta Server occurs about two times per working day. This does not include data downloading activities by collaborators or repeated data downloading afterwards for various reasons. During the same period, MIMI cumulated 1.2 terabytes of fresh imaging data, which translates to a data acquisition rate of 3 gigabytes per working day.

The distribution of accrued content objects during the 18-month period is presented in Table 1. With respect to the anticipated capacity, the Meta Server should be able to handle over 1,000 registered users, 500 projects, 1,000 groups, and 10,000 sessions. The Data Server is designed to maintain 20 terabytes of online data.

Cost–benefit Analysis

It may seem obvious that using a highly integrated system such as MIMI would bring substantial benefits that can be translated to savings in expenditure for CCIR. However, a more precise analysis of the savings using a cost–benefit method involves substantial work. In addition to figuring out the intricacies behind the prior (status quo) practice, we also needed to gather the corresponding cost estimates. A difficult part of the cost–benefit analysis involves the accurate and realistic estimation of the time spent on tasks with the status quo. We caution the reader that, although we tried to get as precise an estimation as we can, there are inherent reasons for some of the estimated figures to be based on rules of the thumb only.

In carrying out the cost–benefit analysis, we followed some existing examples in the literature. These include Grady's analysis of an integrated

Table 1. MIMI's Content Object and Data Statistics

Content Type	Sizes
Registered users	150
Projects	125
Groups	120
Sessions	1,600
Acquired images	1.2TB

telemental health care service for the military,[11] Wang et al.'s cost–benefit analysis of electronic patient medical records,[12] and Erdogmus' approaches of cost–benefit analysis of software development.[13]

Our cost–benefit analysis has focused on directly accountable tasks from the view of the CCIR. This is an underestimate because all the users of the MIMI system receive a fraction of similar benefits on a regular basis as well.

Financial Benefits

Three main tasks have been used for the cost–benefit analysis: scheduling, data distribution, and performance statistics compilation.

Session scheduling. The status quo procedure for scheduling imaging sessions involves three steps:

1. A researcher contacts a CCIR staff member using e-mail or phone.
2. The researcher and staff member work out an amenable time.
3. The staff member schedules an imaging session for the researcher and sends out a notification.

We further analyze each step to estimate the administrative time spent for scheduling an imaging session. The first step is the responsibility of the researcher and does not occupy administrative time. During the second step of the process, the CCIR staff member communicates with the researcher and searches a calendar system for open time slots (the CCIR used Microsoft Outlook Calendar for scheduling imaging sessions). We estimate that the second step takes about 2 min of a staff member's administrative time. This time is obtained as the average of the estimates for phone and for e-mail communication. The third step involves the entry of pertinent data into the calendar system by the staff member. We estimate that the third step uses an additional 0.5 min on average because a valid time slot is already determined in step two. In total, we estimate that scheduling an imaging session takes approximately 2.5 min of a staff member's time. Based on Usage analysis, assuming that that the CCIR averages about four imaging sessions per working day, this translates to an estimated 10 min of administrative time. Assuming the lower end of 260 working days per year and $18 per hour for a low-level administrative staff, the CCIR's annual cost for the low-level administrative staff would be $780. However, a low-level administrative staff cannot handle all the responsibilities of scheduling. A high-level scientific staff member with in-depth knowledge of the imaging systems is involved in final decision making to oversee scheduling management, resolve scheduling conflict and manage the data distribution. This cost is combined in the data distribution cost.

Data distribution. Based on the prior practice at CCIR, a high-level scientific staff member spends half of the time to oversee scheduling and manage data distribution based on file sharing. During a typical working day, the high-level scientific staff member is inundated with requests for rescheduling. He must also set up user accounts for data distribution via file sharing. Users who do not have direct access to the CCIR network are an additional burden for the high-level staff member because their PCs require time-consuming updates to access the CCIR network. The staff member's salary is around $90,000, and the adjusted annual cost will be around $45,780.

Compiling performance statistics. Performance assessment and resource usage analysis is essential for justifying the continued investment and funding for a core facility. This has been a time-consuming task usually involving two steps: (1) locating relevant documents in paper or electronic format and (2) going through the documents, extracting the pertinent information, and summarizing the performance statistics. With the status quo and based on the practice that compilation is performed on a monthly basis, this amounts to a full-time job for two administrative staff members. Assuming that that an administrative staff member's salary is approximately $36,000, the performance statistics task with the status quo incurs an annual cost of $72,000.

Using MIMI, performance statistics are compiled automatically. The administrative time needed amounts to logging into MIMI, issuing a performance summary query, and saving the results. Assuming that such queries are performed no more than several times a week, this incurs negligible time for a staff member.

Costs

The costs for using MIMI are of two kinds: nonrecurring and recurring. Nonrecurring costs include the cost for development and implementation. They also cover hardware and software costs. Recurring costs include hardware upgrades and user training. MIMI's development and implementation cost is approximately \$100,000, with \$50,000 for a full-time programmer, and \$50,000 for a half-time supervisor for design and specification. The software cost for MIMI is \$0 because MIMI is built completely on open-source software that requires neither purchasing fees nor licensing costs. MIMI also incurs a hardware cost of approximately \$3,300. The hardware cost includes a primary server computer and installation fees. Assuming that the server computer is replaced every 3 years results in an estimated annual cost of \$1,100. The estimation of the cost for user training is based on the assumption that the CCIR increases its user base by about 30 people annually. We include the 150 who are currently using MIMI in the final training cost. We also assume that the annual salary of the training personnel is approximately \$36,000, training personnel work 2,000 h annually, and a training session lasts about 2 h and trains 10 users. With these assumptions, we calculated the cost of a training session to be \$36 per 10 users (\$36,000 per year/2,000 h per year×2 h per training session), which equals about \$3.60 per user. We then estimated the cost of training MIMI's initial 150 users to be \$540 (\$3.60 per user×150 users). We also determined that training 30 users per year incurs an annual cost of approximately \$108 (\$3.60 per user×30 users).

Summary

Table 2 shows a summary of MIMI's cost–benefit analysis. The annual financial benefits and costs totals are \$117,780 and \$1,208, respectively.

Table 2. A Summary of MIMI's Cost–Benefit Analysis

	Benefit (\$)	Cost (\$)	Occurs Annually
Scheduling and data distribution	45,780.0	–	Y
Performance statistics	72,000.0	–	Y
Development and implementation	–	100,000.0	N
Initial hardware	–	3,300.0	N
Hardware updates	–	1,100.0	Y
Initial training	–	540.0	N
Further training	–	108.0	Y

MIMI also incurs an initial total cost of \$103,840. A very rough formula for the overall financial gain after a period of n years is:

$$F(n) = \$116572n - \$103840$$

With three specific time points as input samples for the formula, we find that foregoing the status quo methods and using MIMI over time periods of 1, 2, and 3 years yields progressive financial benefits of \$12,732, \$129,304, and \$245,876, respectively. About one million dollars can be saved along this trajectory within 10 years. Again, this saving does not account for overhead savings provided by MIMI for the users in data transfer and sharing.

DISCUSSION

Based on our deployment experiences at the CCIR, we find the baseline MIMI to be a viable solution for managing an imaging core facility's scientific and administrative tasks. Administratively, a core facility may use MIMI to manage users and research projects, schedule its equipment, perform billing, and compile statistical performance assessments. Scientifically, a core facility may use MIMI to manage heterogeneous scientific data and disseminate them to researchers. Due to the generality of design considerations, MIMI can be used for other kinds of core facilities, such as proteomics.

The baseline MIMI is envisioned to grow with additional functionalities. One example is an Application Server, which would serve as a hub for managing scientific data in postanalysis form, with rich links to additional local and off-site biological resources. CCIR researchers can currently download scientific data via MIMI's web-based interface. However, they cannot upload or share analyzed results. The Application Server would play a role similar to a Digital Laboratory Notebook, to support research activities more directly.

ACKNOWLEDGMENTS

We would like to thank the following faculty and staff members for feedback, support, and encouragement during the development of this project: Chris Flask, Raymond Muzic, Janet Schiciano, Joan Schenkel, Jeff Duerk, Mark Chance, Stan Gerson, James Jacobberger, Anne Duli, Keiji Takamoto, Adam Troy, and Gurkan Bebek. This project is partially funded by support from NIH K25EB004467, CCIR, Case Center for Proteomics, and Case Comprehensive Cancer Center.

REFERENCES

1. Schultz C, Weissleder R, Pivovarov, M, Mahmood U, Bhandary G, Datta D, Owens S, Hengerer A, Zahlmann G, Brett D, Rechid R, Naraghi M: Molecular imaging portal: new development it platform for imaging, nonimaging, and genomics. Siemens Med Solut Mag 4:71–77, 2005

2. McDonald J, Schadow G, Barnes M, Dexter P, Overhage J, Mamlin B, McCoy J: Open source software in medical informatics—why, how and what. Int J Med Inform 69:175–184, 2003

3. Thiruvathukal G, Laufer K: Plone and content management. Comput Sci Eng 6:88–95, 2004

4. Meyer E: Cascading Style Sheets: The Definitive Guide. Sebastopol: O'Reilly Media, 2000

5. Zhang G-Q, White L, Hesse C, Buchner M, Mehregany M: Roadmap for a departmental website. Educause Q 28:68–72, 2005

6. Cockburn A: Agile Software Development: the Cooperative Game (2nd edition). Reading: Addison-Wesley, 2006

7. Levy M: Web programming in Guide. Softw Pract Exp 28:1581–1603, 1998

8. Richardson J: Bugs in the Web. Comput Secur 16:332–333, 1997

9. Sammon D, Finnegan P: The ten commandments of data warehousing. ACM SIGMIS DB 31:82–91, 2000

10. Eshuis R: Symbolic model checking of UML activity diagrams. ACM Trans Softw Eng Methodol 15:1–38, 2006

11. Grady C: A comparative cost analysis of an integrated military telemental health-care service. Telemed J E-Health 8:293–300, 2002

12. Huang K, Wong S, Pietka E: Medical image informatics infrastructure design and applications. Med Inform 22:279–289, 1997

13. Erdogmus H, Bauldock B: Cost–benefit analysis of software development techniques and practices. In: Companion Conference Proceedings of the 29th International Conference on Software Engineering, 2007, pp 178–179