Bioimage Informatics for Experimental Biology*

Jason R. Swedlow,[1] Ilya G. Goldberg,[2] Kevin W. Eliceiri,[3] and the OME Consortium[4]

[1]Wellcome Trust Centre for Gene Regulation and Expression, College of Life Sciences, University of Dundee, Dundee DD1 5EH, Scotland, United Kingdom; email: jason@lifesci.dundee.ac.uk

[2]Image Informatics and Computational Biology Unit, Laboratory of Genetics, National Institute on Aging, IRP, NIH Biomedical Research Center, Baltimore MD 21224; email: igg@nih.gov

[3]Laboratory for Optical and Computational Instrumentation, University of Wisconsin at Madison, Madison, Wisconsin 53706; email: eliceiri@wisc.edu

[4]**http://openmicroscopy.org/site/about/development-teams**

Annu. Rev. Biophys. 2009. 38:327–46

The *Annual Review of Biophysics* is online at biophys.annualreviews.org

This article's doi:
10.1146/annurev.biophys.050708.133641

1936-122X/09/0609-0327$20.00

*The U.S. Government has the right to retain a nonexclusive, royalty-free license in and to any copyright covering this paper.

Key Words

microscopy, file formats, image management, image analysis, image processing

Abstract

Over the past twenty years there have been great advances in light microscopy with the result that multidimensional imaging has driven a revolution in modern biology. The development of new approaches of data acquisition is reported frequently, and yet the significant data management and analysis challenges presented by these new complex datasets remain largely unsolved. As in the well-developed field of genome bioinformatics, central repositories are and will be key resources, but there is a critical need for informatics tools in individual laboratories to help manage, share, visualize, and analyze image data. In this article we present the recent efforts by the bioimage informatics community to tackle these challenges, and discuss our own vision for future development of bioimage informatics solutions.

Contents

INTRODUCTION

Modern imaging systems have enabled a new kind of discovery in cellular and developmental biology. With spatial resolutions running from millimeters to nanometers, analysis of cell and molecular structure and dynamics is now routinely possible across a range of biological systems. The development of fluorescent reporters, most notably in the form of genetically encoded fluorescent proteins (FPs), combined with increasingly sophisticated imaging systems has enabled direct study of molecular structure and dynamics (6, 52). Cell and tissue imaging assays have scaled to include all three spatial dimensions, a temporal component, and the use of spectral separation to measure multiple molecules such that a single image is now a five-dimensional structure—space, time, and channel. High content screening (HCS) and fluorescence lifetime, polarization, and correlation are all examples of new modalities that further increase the complexity of the modern microscopy dataset. However, multidimensional data acquisition generates a significant data problem: A typical four-year project generates hundreds of gigabytes of images, perhaps on many different proprietary data acquisition systems, making hypothesis-driven research dependent on data management, visualization, and analysis.

HCS: high content screening

Bioinformatics is a mature science that forms the cornerstone of much of modern biology. Modern biologists routinely use genomic databases to inform their experiments.

In fact these databases are well-crafted multilayered applications that include defined data structures, application programming interfaces (APIs), and use standardized user interfaces to enable querying, browsing, and visualization of the underlying genome sequences. These facilities serve as a great model of the sophistication necessary to deliver complex, heterogeneous datasets to bench biologists. However, most genomic resources work on the basis of defined data structures with defined formats and known identifiers that all applications can access (they also employ expert staff to monitor systems and databases, a resource that is rarely available in individual laboratories). There is no single agreed data format, but a defined number are used in various applications, depending on the exact application (e.g., FASTA and EMBL files). These files are accessed through a number of defined software libraries that translate data into defined data structures that can be used for further analysis and visualization. Because a relatively small number of sequence data generation and collation centers exist, standards have been relatively easy to declare and support. Nonetheless, a key to the successful use of these data was the development of software applications, designed for use by bench biologists as well as specialist bioinformaticists, that enabled querying and discovery based on genomic data held by and served from central data resources.

Given this paradigm, the same facility should in principle be available for all biological imaging data (as well as proteomics and, soon, deep sequencing). In contrast to centralized genomics resources, in most cases, these methods are used for defined experiments in individual laboratories or facilities, and the number of image datasets recorded by a single postdoctoral fellow (hundreds to thousands) can easily rival the number of genomes that have been sequenced to date. For the continued development and application of experimental biology imaging methods, it will be necessary to invest in and develop informatics resources that provide solutions for individual laboratories and departmental facilities. Is it possible to deliver flexible, powerful, and usable informatics tools to manage a single laboratory's data that are comparable to that used to deliver genomic sequence applications and databases to the whole community? Why can't the tools used in genomics be immediately adapted to imaging? Are image informatics tools from other fields appropriate for biological microscopy? In this article, we address these questions, discuss the requirements for successful image informatics solutions for biological microscopy, and consider the future directions that these applications must take to deliver effective solutions for biological microscopy.

Application programming interface (API): an interface providing one software program or library easy access to its functionality with full knowledge of the underlying code or data structures

FLEXIBLE INFORMATICS FOR EXPERIMENTAL BIOLOGY

Experimental imaging data are by their very nature heterogeneous and dynamic. The challenge is to capture the evolving nature of an experiment in data structures that by their very nature are specifically typed and static, for later recall, analysis, and comparison. Achieving this goal in imaging applications means solving a number of problems.

Proprietary File Formats

There are over 50 different proprietary file formats (PFFs) used in commercial and academic image acquisition software packages for light microscopy (34). This number significantly increases if electron microscopy, new HCS systems, tissue imaging systems, and other new modes of imaging modalities are included. Regardless of the specific application, almost all store data in their own PFFs. Each of these formats includes the binary data (i.e., the values in the pixels) and the metadata (i.e., the data that describes the binary data). Metadata include physical pixel sizes, time stamps, spectral ranges, and any other measurements or values required to fully define the binary data. Because of the heterogeneity of microscope imaging experiments, there is no agreed upon community specification for a minimal set of metadata (see below). Regardless, the binary data and metadata

Proprietary file formats (PFFs): image file data formats defined and used by individual entities

combined form the full output of the microscope imaging system, and each software application must contend with the diversity of PFFs and continually update its support for changing formats.

Experimental Protocols

Sample preparation, data acquisition methods and parameters, and analysis workflow all evolve during the course of a project, and there are invariably differences in approach even between laboratories doing similar work. This evolution reflects the natural progression of scientific discovery. Recording this evolution (e.g., "What exposure time did I use in the experiment last Wednesday?") and providing flexibility for changing metadata, especially when new metadata must be supported, are critical requirements for any experimental data management system.

Image Result Management

Many experiments only use a single microscope, but the visualization and analysis of image data associated with a single experiment can generate many additional derived files of varying formats. Typically, these are stored on a hard disk using arbitrary directory structures. Thus an experimental result typically reflects the compilation of many different images, recorded across multiple runs of an experiment and associated processed images, analysis outputs, and result spreadsheets. Keeping these disparate data linked so that they can be recalled and examined at a later time is a critical requirement and a significant challenge.

Remote Image Access

Image visualization requires significant computational resources. Many commercial image-processing tools use specific graphics CPU hardware (and thus depend on the accompanying driver libraries). Moreover, they often do not work well when analyzing data across a network connection to data stored on a remote file system. As work patterns move to wireless connections and more types of portable devices, remote access to image visualization tools, coupled with the ability to access and run powerful analysis and processing, will be required.

Image Processing and Analysis

Substantial effort has gone into the development of sophisticated image processing and analysis tools. In genome informatics, the linkage of related but distinct resources [e.g., WormBase (48) and FlyBase (13)] is possible due to the availability of defined interfaces that different resources use to provide access to underlying data. This facility is critical to enable discovery and collaboration—any algorithm developed to ask a specific question should address all available data. This is especially critical as new image methods are developed—an existing analysis tool should not be made obsolete just because a new file format has been developed that it does not read. When scaled across the large number of analysis tool developers, this is an unacceptable code maintenance burden.

Distributed Processing

As the sizes and numbers of images increase, access to larger computing facilities will be routinely required by all investigators. Grid-based data processing is now available for specific analyses of genomic data, but the burden of moving many gigabytes of data even for a single experiment means that distributed computing must also be made locally available, at least in a form that allows laboratories and facilities to access their local clusters or to leverage an investment in multi-CPU, multi-core machines.

Image Data and Interoperability

Strategic collaboration is one of the cornerstones of modern science and fundamentally consists of scientists sharing resources and data with each other. Biological imaging is composed of several specialized subdisciplines—experimental image acquisition, image processing, and image data mining. Each requires its own domain of expertise and specialization, which is justified because each presents unsolved technical challenges as well as ongoing scientific research. For a group specializing in image analysis to make the best use of its expertise, it needs to have access to image data from groups specializing in acquisition. Ideally, this data should comprise current research questions and not historical image repositories that may no longer be scientifically relevant. Similarly, groups specializing in data mining and modeling must have access to image data and to results produced by image processing groups. Ultimately, this drives the development of useful tools for the community and certainly results in synergistic collaborations that enhance each group's advances.

TOWARDS BIOIMAGE INFORMATICS

The delivery of solutions for the problems detailed above requires the development of a new emerging field known as bioimage informatics (45), which includes the infrastructure and applications that enable discovery of insight using systematic annotation, visualization, and analysis of large sets of images of biological samples. For applications of bioimage informatics in microscopy, we include HCS, in which images are collected from arrayed samples and treated with large sets of siRNAs or small molecules (46), as well as large sets of time-lapse images (26), collections of fixed and stained cells or tissues (10, 18), and even sets of generated localization patterns (59) that define specific collections of localization for reference or for analysis. The development and implementation of successful bioimage informatics tools provide enabling technology for biological discovery in several different ways:

- Management: keeping track of data from large numbers of experiments
- Sharing with defined collaborators: allowing groups of scientists to compare images and analytic tools with one another
- Remote access: ability to query, analyze, and visualize without having to connect to a specific file system or use specific video hardware on the user's computer or mobile device
- Interoperability: interfacing of visualization and analysis programs with any set of data, without concern for file format
- Integration of heterogeneous data types: collection of raw data files, analysis results, annotations, and derived figures into a single resource that is easily searchable and browsable.

Bioimage informatics: infrastructure including specifications, software, and interfaces to support experimental biological imaging

BUILDING BY AND FOR THE COMMUNITY

Given these requirements, how should an image informatics solution be developed and delivered? It certainly will involve the development, distribution, and support of software tools that must be acceptable to bench biologists and must work with all the existing commercial and academic data acquisition, visualization, and analysis tools. Moreover, it must support a broad range of imaging approaches and, if at all possible, include the newest modalities in light and electron microscopy, support extensions into clinical research familiar with microscopy (e.g., histology and pathology), and provide the possibility of extension into modalities that do not use visible light (MRI, CT, ultrasound). Because many commercial image acquisition and analysis software packages are already established as critical research tools, all design, development, and testing must assume and expect integration and interoperability. It therefore seems prudent to avoid a traditional commercial approach and make this type of effort community led, using open source models that are now well defined. This does not exclude the possibility of successful commercial ventures being formed to provide bioimage informatics solutions to the experimental biological community, but a community-led, open source approach will be best placed to provide interfaces between all existing academic and commercial applications.

DELIVERING ON THE PROMISE: STANDARDIZED FILE FORMATS VERSUS "JUST PUT IT IN A DATABASE"

In our experience, there are a few commonly suggested solutions for biological imaging. The first is a common, open file format for microscope imaging. A number of specifications for file formats have been presented, including our own (2, 14). Widespread adoption of standardized image data formats has been successful in astronomy (FITS), crystallography (PDB), and clinical imaging (DICOM), where either most of the acquisition software is developed by scientists or a small number of commercial manufacturers adopt a standard defined by the imaging community. Biological microscopy is a highly fractured market, with at least 40 independent commercial providers. This combined with rapidly developing technology platforms acquiring new kinds of data has stymied efforts at establishing a commonly used data standard.

Against this background, it is worth asking whether defining a standardized format for imaging is at all useful and practical. Standardized file formats and minimum data specifications have the advantage of providing a single or, perhaps more realistically, a small number of data structures for the community to contend with. These facilitate interoperability—visualization and analysis tools developed by one lab may be easily used by another. This is an important step for collaboration and allows data exchange—moving a large multidimensional file from one software application to another, or from one lab or center to another. However, standardized formats only satisfy some of the requirements defined above and provide none of the search, query, remote access, or collaboration facilities discussed above, and thus are only a partial solution. However, the expression of a data model in a standardized file format, and especially the development of software that reads and writes that format, is a useful exercise. It tests the modeling concepts, relationships, and requirements (e.g., "If an objective lens is specified, should the numerical aperture be mandatory?") and provides a relatively easy way for the community to access, use, and comment on the data relationships defined by the project. This is an important component of data modeling and standardization and should not be minimized. Moreover, while not providing most of the functionality defined above, standardized formats have the practical value of providing a medium for the publishing and release of data to the scientific community. Unlike gene sequence and protein structure data, there is no requirement for release of images associated with published results, but the availability of standardized formats may facilitate this.

To provide some of the data management features described above, labs might use any number of commercial database products (e.g., Microsoft Access®, FileMaker Pro®) to build customized local databases on commercial foundations. This is certainly a potential solution for individual laboratories, but to date, these local database efforts have not simultaneously dedicated themselves to addressing interoperability, allowing broad support for alternative analysis and visualization tools that were not specifically supported when the database was built. Perhaps most importantly, single lab efforts often emphasize specific aspects of their own research (e.g., the data model supports individual cell lines, but not yeast or worm strains), and the adaptability necessary to support a range of disciplines across biological research, or even their own evolving repertoire of methods and experimental systems, is not included.

In no way does this preclude the development of local or targeted bioimage informatics solutions. In genomics, several community-initiated informatics projects focused on specific resources support the various biological model systems (13, 50, 57). It seems likely that similar projects will grow up around specific bioimage informatics projects, following the models of the Allen Brain Atlas, E-MAGE, and the Cell Centered Database (CCDB) (10, 18, 21). In genomics, there is underlying interoperability between specialized sources—ultimately all of the sequence data as well as the specialized annotation exist in common repositories and formats (e.g., GenBank). Common repositories are not yet feasible for multidimensional image data, but there will be value in linking through the gene identifiers themselves, ontological annotations, or perhaps, localization maps or sets of phenotypic features, once these are standardized (59). Once these links are made to images stored in common formats, distributed storage may effectively accomplish the same thing as centralized storage.

Several large-scale bioinformatics projects related to interoperability between large biological information datasets have emerged, including caBIG, which focuses on cancer research (7); BIRN, which focuses on neurobiology with a substantial imaging component (4); BioSig (44), which provides tools for large-scale data analysis; and myGrid, which focuses on simulation, workflows, and in silico experiments (25). Projects specifically involved in large-scale imaging infrastructure include the Protein Subcellular Location Image Database (PSLID) (16, 24), Bisque (5), CCDB (9, 21), and our own, the Open Microscopy Environment (OME) (31, 55). All these projects were initiated to support the specific needs of the biological systems and experiments in each of the labs driving the development of each project. For example, studies in neuroscience depend on a proper specification for neuroanatomy so that any image and resulting analysis can be properly oriented with respect to the physiological source. In this case, an ontological framework for neuroanatomy is then needed to support and compare the results from many different laboratories (21). A natural progression is a resource that enables sharing of specific images, across many different resolution scales, that are as well defined as possible (9). PSLID is an alternative repository that provides a well-annotated resource for subcellular localization by fluorescence microscopy. In all cases these projects are the result of dedicated, long-term collaboration between computer scientists and biologists, indicating that the challenges presented by this infrastructure development represent the state of the art not only in biology but in computing as well. Many if not most of these projects make use of at least some common software and data models, and although full interoperability is not something that can be claimed today, key members of these projects regularly participate in the same meetings and working groups. In the future, it should be possible for these projects to interoperate to enable, for example, OME software to upload to PSLID or CCDB.

CCDB: Cell Centered Database

PSLID: Protein Subcellular Location Image Database

OME: Open Microscopy Environment

OME: A COMMUNITY-BASED EFFORT TO DEVELOP IMAGE INFORMATICS TOOLS

Since 2000, the Open Microscopy Consortium has been working to deliver tools for image informatics for biological microscopy. Our original vision (55), to provide software tools to enable interoperability between as many image data storage, analysis, and visualization applications as possible, remains unchanged. However, the project has evolved and grown since its founding to encompass a much broader effort and now includes subprojects dedicated to data modeling (37), file format specification and conversion (34, 35), data management (27), and image-based machine learning (29). The Consortium (28) also maintains links with many academic and commercial partners (32). While the challenges of running and maintaining a larger Consortium are real, the major benefits are synergies and feedback that develop when our own project has to use its own updates to data models and file formats. Within the Consortium, there is substantial expertise in data modeling and software development, and we have adopted a series of project management tools and practices to make the project as professional as possible, within the limits of working within academic laboratories. Moreover, our efforts occur within the context of our own image-based research activities. We make no pretense that this samples the full range of potential applications for our specifications and software, just that our

XML: extensible markup language

ideas and work are actively tested and refined before release to the community. Most importantly, the large Consortium means that we can interact with a larger community, gathering requirements and assessing acceptance and new directions from a broad range of scientific applications.

THE FOUNDATION: THE OME DATA MODEL

Since its inception in 2000, the OME Consortium has dedicated itself to developing a specification for the metadata associated with the acquisition of a microscope image. Initially, our goal was to specify a single data structure that would contain spatial, temporal, and spectral components [often referred to as Z, C, T, which together form a 5D image (1)]. This has evolved into specifications for the other elements of the digital microscope system including objective lenses, fluorescence filter sets, illumination systems, and detectors. This effort has been greatly aided by many discussions about configurations and specifications with commercial imaging device manufacturers (32). This work is ongoing, with our current focus being the delivery of specifications for regions-of-interest [based on existing specifications from the geospatial community (42)] and a clear understanding of what data elements are required to properly define a digital microscope image. This process is most efficient when users or developers request updates to the OME data model—the project's Web site (37) accepts requests for new or modified features and fixes.

OME FILE FORMATS

The specification of an open, flexible file format for microscope imaging provides a tool for data exchange between distinct software applications. It is certainly the lowest level of interoperability, but for many situations it suffices in its provision of readable, defined structured image metadata. OME's first specification cast a full 5D image—binary and metadata—in an XML (extensible markup language) file (14). Although conceptually sound, a more pragmatic approach is to store binary data as TIFF images and then link image metadata represented as OME-XML by including it within the TIFF image header or as a separate file (35). To ensure that these formats are in fact defined, we have delivered an OME-XML and OME-TIFF file validator (36) that can be used by developers to ensure files follow the OME-XML specification. As of this writing five commercial companies support these file formats in their software with a "Save as..." option, thus enabling export of image data and metadata to a vendor-neutral format.

SUPPORT FOR DATA TRANSLATION—BIO-FORMATS

PFFs are perhaps the most common informatics challenge faced by bench biologists. Despite the OME-XML and OME-TIFF specifications, PFFs will continue to be the dominant source of raw image for visualization and analysis applications for some time. Because all software must contend with PFFs, the OME Consortium has dedicated its resources to developing a software library that can convert PFFs to a vendor-neutral data structure—OME-XML. This led to the development, release, and continued maintenance of Bio-Formats, a standalone Java library for reading and writing life sciences image formats. The library is general, modular, flexible, extensible, and accessible. The project originally grew out of efforts to add support for file formats to the LOCI VisBio software (40, 49) for visualization and analysis of multidimensional image data, when we realized that the community was in acute need of a broader solution to the problems created by myriad incompatible microscopy formats.

Utility

Over the years we have repeatedly observed software packages reimplement support for the same microscopy formats [i.e., ImageJ (17), MIPAV (22), BioImageXD (3), and many commercial packages]. The vast majority of these efforts focus exclusively on adaptation of formats into each program's specific internal data model; Bio-Formats (34), in contrast, unites popular life sciences file formats under a broad, evolving data specification provided by the OME data model. This distinction is critical: Bio-Formats does not adapt data into structures designed for any specific visualization or analysis agenda, but rather expresses each format's metadata in an accessible data model built from the ground up to encapsulate a wide range of scientifically relevant information. We know of no other effort within the life sciences with as broad a scope as Bio-Formats and dedicated toward delivering the following features.

Modularity

The architecture of the Bio-Formats library is split into discrete, reusable components that work together but are fundamentally separable. Each file format reader is implemented as a separate module extending a common IFormatReader interface; similarly, each file format writer module extends a common IFormatWriter interface. Both reader and writer modules utilize the Bio-Formats MetadataStore API to work with metadata fields in the OME Data Model. Shared logic for encoding and decoding schemes (e.g., JPEG and LZW) are structured as part of the Bio-Formats codec package, so that future readers and writers that need those same algorithms can leverage them without reimplementing similar logic or duplicating any code.

When reading data from a dataset, Bio-Formats provides a tiered collection of reader modules for extracting or restructuring various types of information from the dataset. For example, a client application can instruct Bio-Formats to compute minimum and maximum pixel values using a MinMaxCalculator, combine channels with a ChannelMerger, split them with a ChannelSeparator, or reorder dimensional axes with a DimensionSwapper. Performing several such operations can be accomplished merely by stacking the relevant reader modules one on top of the other. Several auxiliary components are also provided; the most significant are a caching package for intelligent management of image planes in memory when storage requirements for the entire dataset would be too great, and a suite of graphical components for common tasks such as presenting the user with a file chooser dialog box or visualizing hierarchical metadata in a tree structure.

Flexibility

Bio-Formats has a flexible metadata API, built in layers over the OME Data Model itself. At the lowest level, the OME Data Model is expressed as an XML schema, called OME-XML, that is continually revised and expanded to support additional metadata fields. An intermediate layer known as the OME-XML Java library is produced using code generation techniques, which provides direct access to individual metadata fields in the OME-XML hierarchy. The Bio-Formats metadata API, which provides a simplified, flattened version of the OME Data Model for flexible implementation by the developer, leverages the OME-XML Java library layer and is also generated automatically from underlying documents to reduce errors in the implementation.

Extensibility

Adding a new metadata field to the data model is done at the lowest level, to the data model itself via the OME-XML schema. The supporting code layers—both the OME-XML Java library and the Bio-Formats metadata API—are programmatically regenerated to include the addition. The only remaining task is to add a small amount of code to each file format reader mapping the original data field into the appropriate location within the standardized OME Data Model.

Although the OME Data Model specifically targets microscopy data, in general, the Bio-Formats model of metadata extensibility is ideal for adaptation to alternative data

OMERO: OME Remote Objects

RDMS: relational database management system

models unrelated to microscopy. By adopting a similar pattern for the new data model, and introducing code generation layers corresponding to the new model, the Bio-Formats infrastructure could easily support additional branches of multidimensional scientific imaging data. In the future the Bio-Formats infrastructure will provide significant interoperability between the multiple established data models at points where they overlap by establishing a common base layer between them.

Bio-Formats is written in Java so that the code can execute on a wide variety of target platforms, and code and documentation for interfacing Bio-Formats with a number of different tools including ImageJ, MATLAB, and IDL are available (34). We provide documentation on how to use Bio-Formats both as an end user and as a software developer, including hints on leveraging Bio-Formats from other programming environments such as C++, Python, or a command shell. We have successfully integrated Bio-Formats with native acquisition software written in C++ using ICE middleware (58).

DATA MANAGEMENT APPLICATIONS: OME AND OMERO

Data management is a critical application for modern biological discovery, and in particular necessary for biological imaging because of the large heterogeneous datasets generated during data acquisition and analysis. We define data management as the collation, integration, annotation, and presentation of heterogeneous experimental and analytic data in ways that enable the physical, temporal, and conceptual relationships in experimental data to be captured and represented to users. The OME Consortium has built two data management tools—the original OME Server (29) and the recently released OME Remote Objects (OMERO; a port of the basic image data management functionality to a Java enterprise application) application platform (30). Both applications are now heavily used worldwide, but our development focus has shifted from the OME Server toward OMERO, and that is where most future advances will occur.

The OME data management applications are specifically designed to meet the requirements and challenges described above, enabling the storage, management, visualization, and analysis of digital microscope image data and metadata. The major focus of this work is not on creating novel analysis algorithms, but instead on development of a structure that ultimately allows any application to read and use any data associated with or generated from digital microscope images.

A fundamental design concept in the OME data management applications is the separation of image storage, management, analysis, and visualization functions between a lab's or imaging facility's server and a client application (e.g., Web browser or Java user interface). This concept mandates the development of two facilities: a server that provides all data management, access control, and storage, and a client that runs on a user's desktop workstation or laptop and that provides access to the server and the data via a standard Internet connection. The key to making this strategy work is judicious choice of the functionality placed on client and server to ensure maximal performance.

The technical design details and principles of both systems have recently been described (23) and are available online (39). In brief, both the OME Server and the OMERO platform (**Figure 1**) use a relational database management system (RDMS) [PostgreSQL (47)] to provide all aspects of metadata management and an image repository to house all the binary pixel data. Both systems then use a middleware application to interact with the RDMS and read and write data from the image repository. The middleware applications include a rendering engine that reads binary data from the image repository and renders it for display by the client, and if necessary, compresses the image to reduce the bandwidth requirements for transferring across a network connection to a client. The result is access to high-performance data visualization, management,

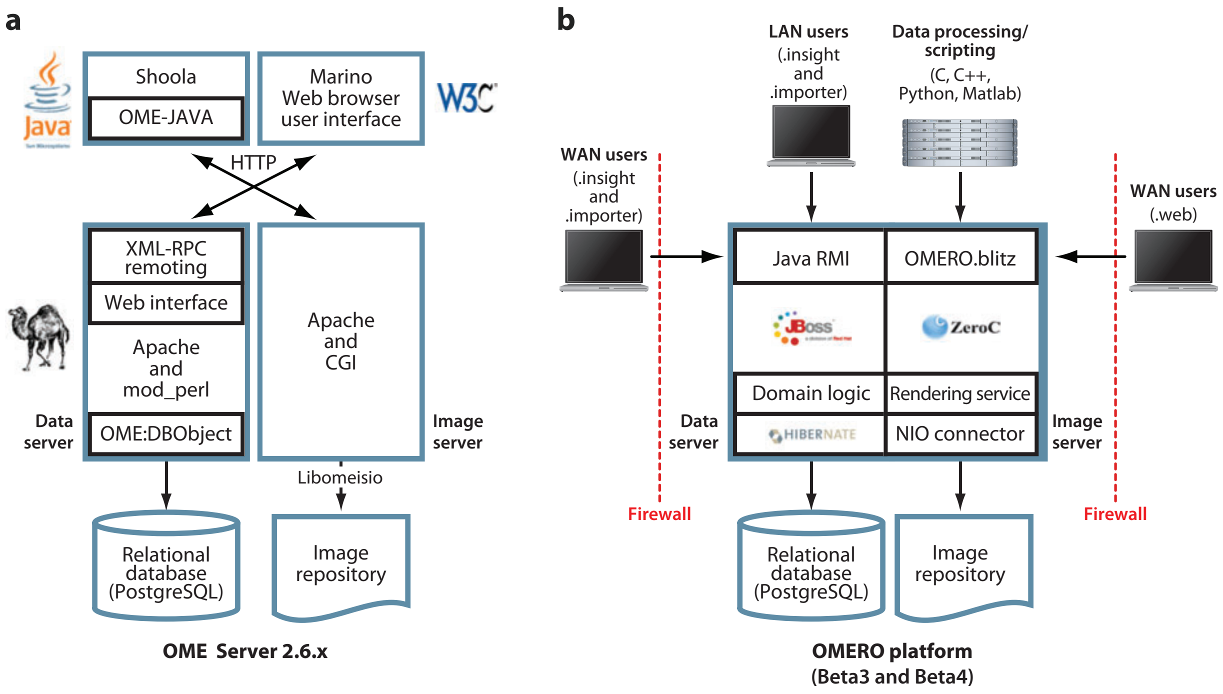

Figure 1

Architecture of the OME and OMERO servers and client applications. (*a*) Architecture of the OME Server, built using Perl for most of the software code and an Apache Web server. The main client application for the server is a Web browser-based interface. (*b*) The architecture of the OMERO platform, including OMERO.server and the OMERO clients. OMERO is based on the JBOSS JavaEE framework, but it also includes an alternative remoting architecture called ICE (58). For more details, see Reference 39.

and analysis in a remote setting. Both the OME Server (**Figure 1*a***) and OMERO (**Figure 1*b***) also provide well-developed data querying facilities to access metadata, annotations, and analytics from the RDMS. For user interfaces, the OME Server includes a Web browser-based interface that provides access to image, annotation, analytics, and visualization and also a Java interface (OME-JAVA) and remote client (Shoola) to support access from remote client applications. OMERO includes separate Java-based applications for uploading data to an OMERO server (OMERO.importer), for visualizing and managing data (OMERO.insight), and for Web browser-based server administration (OMERO.webadmin).

The OME Server has been installed in hundreds of facilities worldwide; however, after significant development effort it became clear that the application, which we worked on for five years (2000–2005), had three major flaws. (*a*) The installation was too complex, and too prone to failure. (*b*) Our object-relational mapping library ("DBObject") was all custom code, developed by OME, and required significant code maintenance effort to maintain compatibility with new versions of Linux and Perl (**Figure 1*a***). Support for alternative RDMSs (e.g., Oracle®) was possible in principle but required significant work. (*c*) The data transport mechanisms available to us in a Perl-based architecture amounted to XML-RPC and SOAP. Although totally standardized and promoting interoperability, this mechanism, with its requirement for serialization/deserialization of large data objects, was too slow for working with remote client applications—simple queries with well-populated databases could take minutes to transfer from server to client.

With work, problem *a* became less of an issue, but problems *b* and *c* remained significant fundamental barriers to delivery of a great image informatics application to end users. For these reasons, we initiated work on OMERO. In taking on this project, it was clear that the code maintenance burden needed to be substantially reduced, the system must be simple to install, and the performance of the remoting system must be significantly improved. A major design goal was the reduction of self-written code through the reuse of existing middleware and tools where possible. In addition, OMERO must support as broad a range of client applications as possible, enabling the development of new user interfaces, as well as a wide range of data analysis applications.

We based the initial implementation of OMERO's architecture (**Figure 1*b***) on the JavaEE5 specification, as it appeared to have wide uptake, clear specifications, and high performance libraries in active development from a number of projects. A full specification and description of OMERO.server is available (23). The architecture follows accepted standards and consists of services implemented as EJB3 session beans (53) that make use of Hibernate (15), a high-performance object-relational mapping solution, for metadata retrieval from the RDMS. Connection to clients is via Java Remote Method Invocation (Java RMI) (54). All released OMERO remote applications are written in Java and cross-platform. OMERO.importer uses the Bio-Formats library to read a range of file formats and load the data into an OMERO.server, along with simple annotations and assignment to the OME Project-Dataset-Image experimental model (for demonstrations, see Reference 33). OMERO.insight includes facilities for managing, annotating, searching, and visualizing data stored in an installation of OMERO.server. OMERO.insight also includes simple line and region-of-interest measurements and thus supports the simplest forms of image analysis.

OMERO ENHANCEMENTS: BETA3 AND BETA4

Through 2007, the focus of the OMERO project has been on data visualization and management, all the while laying the infrastructure for data analysis. With the release of OMERO3-Beta2, we began adding functionality that has the foundation for delivering a fully developed image informatics framework. In this section, we summarize the major functional enhancements that are being delivered in OMERO-Beta3 (released June 2008) and OMERO-Beta4 (released February 2009). Further information on all the items described below is available at the OMERO documentation portal (39).

OMERO.blitz

Starting with OMERO-Beta3, we provided interoperability with many different programming environments. We chose an ICE-based framework (58) rather than the more popular Web services–based GRID approaches because of the absolute performance requirements we had for the passage of large binary objects (image data) and large data graphs (metadata trees) between server and client. Our experience using Web services and XML-based protocols with the Shoola remote client and the OME Server showed that Web services, while standardized in most genomic applications, were inappropriate for client-server transfer of the much larger data graphs we required. Most importantly, the ICE framework provided immediate support for multiple programming environments (C, C++, and Python are critical for our purposes) and a built-in distribution mechanism [IceGRID (58)] that we have adapted to deliver OMERO.grid (39), a process distribution system. OMERO.blitz is three to four times faster than Java RMI and we are currently examining migrating our Java API and the OMERO clients from JBOSS to OMERO.blitz. This framework provides substantial flexibility—interacting with data in OMERO can be as simple as starting the Python interpreter and interacting with OMERO via the console. Most importantly, this strategy forms the foundation for our future work as we can now leverage the advantages and existing functionality in cross-platform Java, native C and C++, and scripted Python for rapidly expanding the functionality in OMERO.

Structured Annotations

Beginning with OMERO-Beta3, users can attach any type of data to an image or other OMERO data container—text, URL, or other data files (e.g., .doc, .pdf, .xls, .xml) providing essentially the same flexibility as email attachments. The installation of this facility followed feedback from users and developers concerning the strategy for analysis management built into the OME Server. The underlying data model supported hard semantic typing in which each analysis result was stored in relational tables with names that could be defined by the user (23, 55). This approach, although conceptually desirable, proved too complex and burdensome. As an alternative, OMERO uses Structured Annotations to store any kind of analysis result as untyped data, defined only by a unique name to ensure that multiple annotations are easily distinguished. The data are not queryable by standard SQL, but any text-based file can be indexed and therefore found by users. Interestingly, Bisque has implemented a similar approach (5), enabling tags with defined structures that are otherwise completely customized by the user. In both cases, whether this flexible strategy provides enough structure to manage large sets of analysis results will have to be assessed.

OMERO.search

As of OMERO-Beta3, OMERO includes a text-indexing engine based on Lucene (19), which can be used to provide indexed-based searches for all text-based metadata in an OMERO database. This includes metadata and annotations stored within the OMERO database and also any text-based documents or results stored as Structured Annotations.

OMERO.java

As of OMERO-Beta3, we have released OMERO.java, which provides access for all external Java applications via the OMERO.blitz interface. As a first test of this facility, we are using analysis applications written in MATLAB as client applications to read from and write to OMERO.server. As a demonstration of the utility of this library, we have adapted the popular open source MATLAB-based image analysis tool CellProfiler (8) to work as a client of OMERO, using the MATLAB Java interface.

OMERO.editor

In OMERO-Beta3, we also released OMERO.editor, a tool to help experimental biologists define their own experimental data models and, if desired, use other specified data models in their work. It allows users to create a protocol template and to populate this with experimental parameters. This creates a complete experimental record in one XML file, which can be used to annotate a microscope image or exchanged with other scientists. OMERO.editor supports the definition of completely customized experimental protocols but also includes facilities to easily import defined data models [e.g., MAGE-ML (56) and OME-XML (14)] and support for all ontologies included in the Ontology Lookup Service (11).

OMERO.web

Staring with OMERO-Beta4, we will release a Web browser-based client for OMERO.server. This new client is targeted specifically to truly remote access (different country, limited bandwidth connections), especially where collaboration with other users is concerned. OMERO.web includes all the standard functions for importing, managing, viewing, and annotating image data. However, a new function is the ability to share specific sets of data with another user on the system—this allows password-protected access to a specific set of data that can initiate or continue data sharing between two lab members or two collaborating scientists. OMERO.web also supports a publish function, in which a defined set of data is published to the world via a public URL. OMERO.web uses the Python API in OMERO.blitz for access to OMERO.server using the Django framework (12).

OMERO.scripts

In OMERO-Beta4, we will extend the analysis facility provided by OMERO.java to provide a scripting engine, based on Python scripts and the OMERO.blitz interface. OMERO.scripts is a scripting engine that reads and executes functions cast in Python scripts. Scripts are passed to processors specified by OMERO.grid that can be on the local server or on networked computing facilities. This is the facility that will provide support for analysis of large image sets or of calculations that require simple linear or branched workflows.

OMERO.fs

Finally, a fundamental design principle of OMERO.server is the presence of a single image repository for storing binary image data that is tightly integrated with the server application. This is the basis of the import model, which is the only way to get image data into an OMERO.server installation—data are uploaded to the server, and binary data are stored in the single image repository. In many cases, as the storage space required expands, multiple repositories must be supported. Moreover, data import takes significant time and, especially with large datasets, can be prohibitive. A solution to this involves using the OMERO.blitz Python API to access the file system search and notification facilities that are now provided as part of the Windows, Linux, and OS X operating systems. In this scenario, an OMERO client application, OMERO.fs, sits between the file system and OMERO.blitz and provides a metadata service that scans user-specified image folders or file systems and reads image metadata into an OMERO relational database using PFF translation provided by Bio-Formats. As the coverage of Bio-Formats expands, this approach means that essentially any data can be loaded into an OMERO.server instance.

WORKFLOW-BASED DATA ANALYSIS: WND-CHARM

WND-CHARM is an image analysis algorithm based on pattern recognition (43). It relies on supervised machine learning to solve image analysis problems by example rather than by using a preconceived perceptual model of what is being imaged. An advantage of this approach is its generality. Because the algorithms used to process images are not task specific, they can be used to process any image regardless of the imaging modality or the image's subject. Similar to other pattern recognition algorithms, WND-CHARM first decomposes each image to a set of predefined numeric image descriptors. Image descriptors include measures of texture, factors in polynomial decompositions, and various statistics of the image as a whole, as well as measurements and distribution of high-contrast objects in the image. The algorithms that extract these descriptors (features) operate on both the original image pixels as well as transforms of the original pixels (Fourier, wavelet, etc). Together, there are 17 independent algorithms comprising 53 computational nodes (algorithms used along specific upstream data flows), with 189 links (data flows) producing 1025 numeric values (**Figure 2**). Although the entire set of features can be modeled as a single algorithm, this set is by no means complete and will grow to include other algorithms that extract both more specific and more general image content. The advantage of modeling this complex workflow as independently functional units is that new units can be easily added to the existing ones. This workflow model is therefore more useful to groups specializing in pattern recognition. Conversely, a monolithic

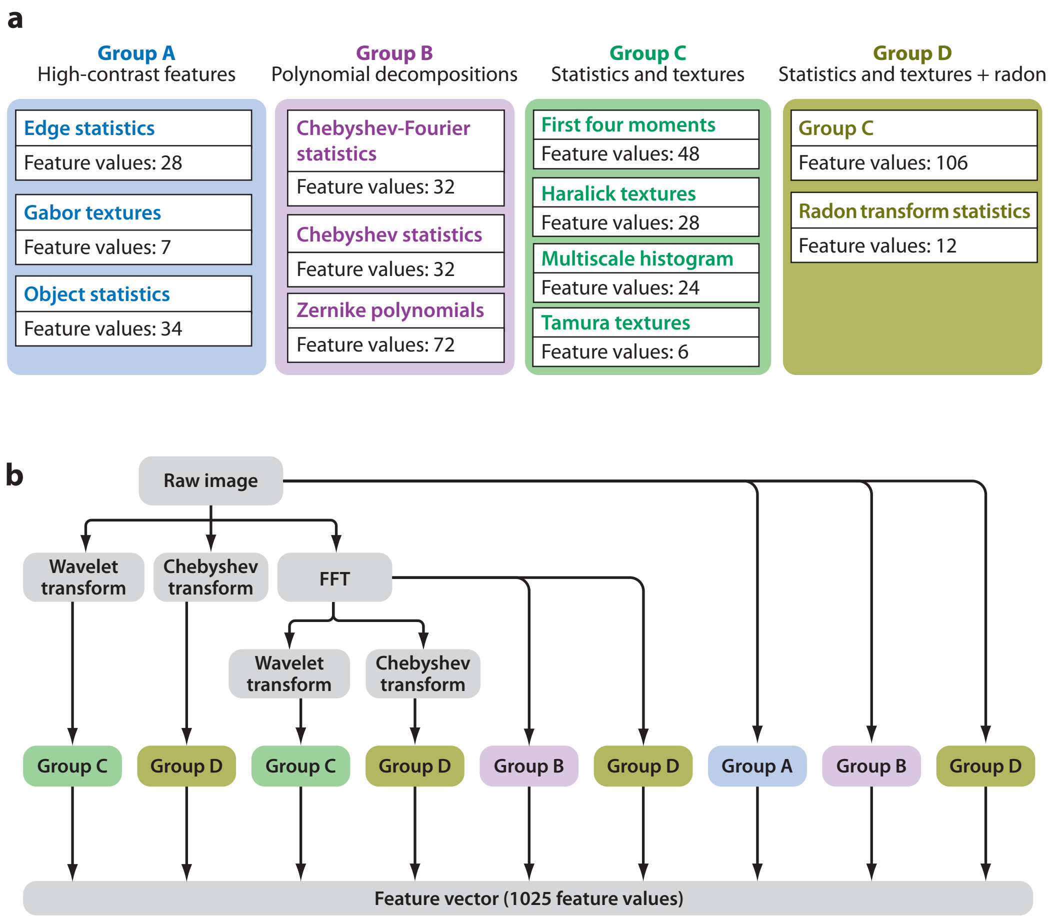

Figure 2

Workflows in WND-CHARM. (*a*) List of feature types calculated by WND-CHARM. (*b*) Workflow of feature calculations in WND-CHARM. Note that different feature groups use different sets of processing tools.

representation of this workflow is probably more practical when implemented in a biology lab that would use a standard set of image descriptors applied to various imaging experiments. In neither case, however, should anyone be particularly concerned with what format was used to capture these images, or how they are represented in a practical imaging system. WND-CHARM is an example of a highly complex image-processing workflow and as such represents an important application for any system capable of managing workflows and distributed processing for image analysis. Currently, the fully modularized version of WND-CHARM runs only on the OME Server. In the near future, the monolithic version of WND-CHARM (51) will be implemented using OMERO.blitz.

The raw results from a pattern recognition application are annotations assigned to whole images or image regions. These annotations are probabilities (or simply scores) that the image or region-of-interest belongs to a previously defined training set. In a dose-response experiment, for example, the training set may consist of control doses defining a standard curve, and the experimental images would be assigned an equivalent dose by the pattern recognition algorithm. Whereas the original experiment may be concerned with characterizing a collection of chemical compounds, the same image data could be analyzed in the context of a different set of training images—one defined by RNA interference, for example. When using these algorithms our group has found that performing these in silico experiments to reprocess existing image data in different contexts can be fruitful.

USABILITY

All the functionality discussed above must be built into OMERO.server and then delivered in a functional, usable fashion within the OMERO client applications OMERO.importer, OMERO.insight, and OMERO.web. This development effort is achieved by the OMERO development team and is invariably an iterative process that requires testing by our local community, as well as sampling feedback from the broader community of users. Therefore, the OMERO project has made software usability a priority throughout the project. A key challenge for the OME Consortium has been to improve the quality of the end user (i.e., the life scientist at their bench) experience. The first versions of OME software, the OME Server, provided substantial functionality but never received wide acceptance, despite dedicated work, mostly because its user interfaces were too complicated and the developed code, while open and available, was too complex for other developers to adopt and extend. In response to this failure, we initiated the Usable Image project (41) to apply established methods from the wider software design community, such as user-centered design and design ethnography (20), to the OME development process. Our goals were to initially improve the usability and accessibility of the OMERO client software and to provide a paradigm useful for the broader e-science and bioinformatics communities. The result of this combined usability and development effort has been a high level of success and acceptance of OMERO software. A wholly unanticipated outcome has been the commitment to the user-centered design process by both users and developers. The investment in iterative, agile development practice has produced rapid, substantial improvements that the users appreciate, which in turn makes them more enthusiastic about the software. On the other hand, the developers have reliable, well-articulated requirements that, when implemented in software, are rewarded with more frequent use. This positive-feedback loop has transformed our development process and made usability analysis a core part of our development cycles. It has also forced a commitment to the development of usable code—readable, well-documented, tested, and continuously integrated—and the provision of up-to-date resources defining architecture and code documentation (38, 39).

SUMMARY AND FUTURE IMPACT

In this article we have focused on the OME Consortium's efforts (namely OME-XML, Bio-Formats, OMERO, and WND-CHARM), as we feel they are representative of the community-wide attempts to address many of the most pressing challenges in bioimaging informatics. While OME is committed to developing and releasing a complete image informatics infrastructure focused on the needs of the end user bench biologist, we are at least equally committed to the concept that beyond our software, our approach is part of a critical shift in how the challenges of data analysis, management, sharing, and visualization have been traditionally addressed in biology. In particular the OME Consortium has put an emphasis on flexibility, modularity, and inclusiveness that targets not only the bench biologist but also importantly the informatics developer to help ensure maximum implementation of and penetration into the bioimaging community. Key to this has been a dedication to allowing the biologist to retain and capture all available metadata and binary data from disparate sources, including proprietary ones, to map these data to a flexible data model, and to analyze these data in whatever environment he or she chooses. This ongoing effort requires an interdisciplinary approach that combines concepts from traditional bioinformatics, ethnography, computer science, and data visualization. It is our intent and hope that the bioimage informatics infrastructure that is developed by the OME Consortium will continue to have utility for its principal target community of experimental bench biologists, and also serve as a collaborative framework for developers and researchers from other closely related fields who might want to adopt the methodologies and code-based approaches for informatics challenges that exist in other communities. Interdisciplinary collaboration between biologists, physicists, engineers, computer scientists, ethnographers, and software developers is absolutely necessary for the successful maturation of the bioimage informatics community, and it will play an even larger role as this field evolves to fully support the continued evolution of imaging in modern experimental biology.

SUMMARY POINTS

1. Advances in digital microscopy have driven the development of a new field, bioimage informatics. This field encompasses the storage, querying, management, analysis, and visualization of complex image data from digital imaging systems used in biology.
2. Although standardized file formats have often been proposed to be sufficient to provide the foundation for bioimage informatics, the prevalence of PFFs and the rapidly evolving data structures needed to support new developments in imaging make this impractical.
3. Standardized APIs and software libraries enable interoperability, which is a critical unmet need in cell and developmental biology.
4. A community-driven development project is best placed to define, develop, release, and support these tools.
5. A number of bioimage informatics initiatives are underway, and collaboration and interaction are developing.
6. The OME Consortium has released specifications and software tools to support bioimage informatics in the cell and developmental biology community.
7. The next steps in software development will deliver increasingly sophisticated infrastructure applications and should deliver powerful data management and analysis tools to experimental biologists.

FUTURE ISSUES

1. Further development of the OME Data Model must keep pace with and include advances in biological imaging, with a particular emphasis on improving support for image analysis metadata and enabling local extension of the OME Data Model to satisfy experimental requirements with good documentation and examples.
2. Development of Bio-Formats to include as many biological image file formats as possible and extension to include data from non-image-based biological data.

3. Continue OMERO development as an image management system with a particular emphasis on ensuring client application usability and the provision of sophisticated image visualization and analysis tools.
4. Support both simple and complex analysis workflow as a foundation for common use of data analysis and regression in biological imaging.
5. Drive links between the different bioimage informatics enabling transfer of data between instances of the systems so that users can make use of the best advantages of each.

DISCLOSURE STATEMENT

The authors are not aware of any biases that might be perceived as affecting the objectivity of this review.

ACKNOWLEDGMENTS

JRS is a Wellcome Trust Senior Research Fellow and work in his lab on OME is supported by the Wellcome Trust (Ref 080087 and 085982), BBSRC (BB/D00151X/1), and EPSRC (EP/D050014/1). Work in IGG's lab is supported by the National Institutes of Health. The OME work in KWE's lab is supported by NIH grants R03EB008516 and R01EB005157.

LITERATURE CITED

1. Andrews PD, Harper IS, Swedlow JR. 2002. To 5D and beyond: quantitative fluorescence microscopy in the postgenomic era. *Traffic* 3:29–36
2. Berman J, Edgerton M, Friedman B. 2003. The tissue microarray data exchange specification: a community-based, open source tool for sharing tissue microarray data. *BMC Med. Inform. Decis. Making* 3:5
3. BioImageXD. 2008. *BioImageXD—open source software for analysis and visualization of multidimensional biomedical data*. **http://www.bioimagexd.net/**
4. BIRN. 2008. **http://www.nbirn.net/**
5. Centre for Bio-Image Informatics. 2008. *Bisque database*. **http://www.bioimage.ucsb.edu/downloads/Bisque%20Database**
6. Bullen A. 2008. Microscopic imaging techniques for drug discovery. *Nat. Rev. Drug Discov.* 7:54–67
7. caBIG Community Website. **http://cabig.nci.nih.gov/**
8. Carpenter A, Jones T, Lamprecht M, Clarke C, Kang I, et al. 2006. CellProfiler: image analysis software for identifying and quantifying cell phenotypes. *Genome Biol.* 7:R100
9. Cell Centered Database. 2008. **http://ccdb.ucsd.edu/CCDBWebSite/**
10. Christiansen JH, Yang Y, Venkataraman S, Richardson L, Stevenson P, et al. 2006. EMAGE: a spatial database of gene expression patterns during mouse embryo development. *Nucleic Acids Res.* 34:D637–41
11. Cote RG, Jones P, Martens L, Apweiler R, Hermjakob H. 2008. The Ontology Lookup Service: more data and better tools for controlled vocabulary queries. *Nucleic Acids Res.* 36:W372–76
12. Django Project. 2008. **http://www.djangoproject.com/**
13. Drysdale R, FlyBase Consortium. 2008. FlyBase: a database for the *Drosophila* research community. *Methods Mol. Biol.* 420:45–59
14. Goldberg IG, Allan C, Burel J-M, Creager D, Falconi A, et al. 2005. The Open Microscopy Environment (OME) data model and XML file: open tools for informatics and quantitative analysis in biological imaging. *Genome Biol.* 6:R47
15. hibernate.org. 2008. **http://www.hibernate.org/**

16. Huang K, Lin J, Gajnak JA, Murphy RF. 2002. Image content-based retrieval and automated interpretation of fluorescence microscope images via the protein subcellular location image database. *Proc. 2002 IEEE Int. Symp. Biomed. Imaging* 325–28
17. Image J. 2008. **http://rsbweb.nih.gov/ij/**
18. Lein ES, Hawrylycz MJ, Ao N, Ayres M, Bensinger A, et al. 2007. Genome-wide atlas of gene expression in the adult mouse brain. *Nature* 445:168–76
19. Apache Lucene. 2008. *Overview*. **http://lucene.apache.org/java/docs/**
20. Macaulay C, Benyon D, Crerar A. 2000. Ethnography, theory and systems design: from intuition to insight. *Int. J. Hum. Comput. Stud.* 53:35–60
21. Martone ME, Tran J, Wong WW, Sargis J, Fong L, et al. 2008. The Cell Centered Database project: an update on building community resources for managing and sharing 3D imaging data. *J. Struct. Biol.* 161:220–31
22. MIPAV. 2008. **http://mipav.cit.nih.gov/**
23. Moore J, Allan C, Burel J-M, Loranger B, MacDonald D, et al. 2008. Open tools for storage and management of quantitative image data. *Methods Cell Biol.* 85:555–70
24. Murphy RF. 2008. *Murphy lab—Imaging services—PSLID*. **http://murphylab.web.cmu.edu/services/PSLID/**
25. myGrid. 2008. **http://www.mygrid.org.uk/**
26. Neumann B, Held M, Liebel U, Erfle H, Rogers P, et al. 2006. High-throughput RNAi screening by time-lapse imaging of live human cells. *Nat. Methods* 3:385–90
27. OME Consortium. 2008. *Data management*. **http://www.openmicroscopy.org/site/documents/data-management**
28. OME Consortium. 2008. *Development teams*. **http://openmicroscopy.org/site/about/development-teams**
29. OME Consortium. 2008. *OME server*. **http://openmicroscopy.org/site/documents/data-management/ome-server**
30. OME Consortium. 2008. *OMERO platform*. **http://openmicroscopy.org/site/documents/data-management/omero**
31. OME Consortium. 2008. **http://openmicroscopy.org/site**
32. OME Consortium. 2008. *Partners*. **http://openmicroscopy.org/site/about/partners**
33. OME Consortium. 2008. *Videos for Quicktime*. **http://openmicroscopy.org/site/videos**
34. OME Consortium. 2008. *OME at LOCI—software—Bio-formats library*. **http://www.loci.wisc.edu/ome/formats.html**
35. OME Consortium. 2008. *OME at LOCI—OME-TIFF—overview and rationale*. **http://www.loci.wisc.edu/ome/ome-tiff.html**
36. OME Consortium. 2008. *OME validator*. **http://validator.openmicroscopy.org.uk/**
37. OME Consortium. 2008. **http://ome-xml.org/**
38. OME Consortium. 2008. *Continuous build & integration system for the Open Microscopy Environment project*. **http://hudson.openmicroscopy.org.uk**
39. OME Consortium. 2008. *ServerDesign*. **http://trac.openmicroscopy.org.uk/omero/wiki/ServerDesign**
40. OME Consortium. 2008. *VisBio—introduction*. **http://www.loci.wisc.edu/visbio/**
41. OME Consortium. 2008. *The usable image project*. **http://usableimage.org**
42. OpenGIS$^{®}$. 2008. *Standards and specifications*. **http://www.opengeospatial.org/standards**
43. Orlov N, Shamir L, Macura T, Johnston J, Eckley DM, Goldberg IG. 2008. WND-CHARM: multi-purpose image classification using compound image transforms. *Pattern Recognit. Lett.* 29:1684–93
44. Parvin B, Yang Q, Fontenay G, Barcellos-Hoff MH. 2003. BioSig: An imaging and informatic system for phenotypic studies. *IEEE Trans. Syst. Man Cybern. B* 33:814–24
45. Peng H. 2008. Bioimage informatics: a new area of engineering biology. *Bioinformatics* 24:1827–36
46. Pepperkok R, Ellenberg J. 2006. High-throughput fluorescence microscopy for systems biology. *Nat. Rev. Mol. Cell Biol.* 7:690–96
47. PostgreSQL. 2008. **http://www.postgresql.org/**
48. Rogers A, Antoshechkin I, Bieri T, Blasiar D, Bastiani C, et al. 2008. WormBase 2007. *Nucleic Acids Res.* 36:D612–17

49. Rueden C, Eliceiri KW, White JG. 2004. VisBio: a computational tool for visualization of multidimensional biological image data. *Traffic* 5:411–17
50. *Saccharomyces* Genome Database. 2008. **http://www.yeastgenome.org/**
51. Shamir L, Orlov N, Eckley DM, Macura T, Johnston J, Goldberg IG. 2008. Wndcharm—an open source utility for biological image analysis. *Source Code Biol. Med.* 3:13
52. Shaner NC, Campbell RE, Steinbach PA, Giepmans BN, Palmer AE, Tsien RY. 2004. Improved monomeric red, orange and yellow fluorescent proteins derived from *Discosoma* sp. red fluorescent protein. *Nat. Biotechnol.* 22:1567–72
53. Sun Microsystems. 2008. *Enterprise Javabeans Technology*. **http://java.sun.com/products/ejb**
54. Sun Microsystems. 2008. *Remote method invocation home*. **http://java.sun.com/javase/technologies/core/basic/rmi/index.jsp**
55. Swedlow JR, Goldberg I, Brauner E, Sorger PK. 2003. Informatics and quantitative analysis in biological imaging. *Science* 300:100–2
56. Whetzel PL, Parkinson H, Causton HC, Fan L, Fostel J, et al. 2006. The MGED Ontology: a resource for semantics-based description of microarray experiments. *Bioinformatics* 22:866–73
57. WormBase. 2008. **http://www.wormbase.org/**
58. ZeroC^{TM}. 2008. **http://www.zeroc.com/**
59. Zhao T, Murphy RF. 2007. Automated learning of generative models for subcellular location: Building blocks for systems biology. *Cytometry A* 71:978–90

Contents

Annual Review of Biophysics

Volume 38, 2009

An online log of corrections to *Annual Review of Biophysics* articles may be found at http://biophys.annualreviews.org/errata.shtml