OMERO and Bio-Formats 5: Flexible Access to Large Bioimaging Datasets at Scale

Josh Moore[a,b], Melissa Linkert[b], Colin Blackburn[a], Mark Carroll[a], Richard K. Ferguson[a], Helen Flynn[a], Kenneth Gillen[a], Roger Leigh[a], Simon Li[a], Dominik Lindner[a], William J. Moore[a], Andrew J. Patterson[a], Blazej Pindelski[a], Balaji Ramalingam[a], Emil Rozbicki[b], Aleksandra Tarkowska[a], Petr Walczysko[a], Chris Allan[b], Jean-Marie Burel[a], Jason R. Swedlow[a,b]

[a]Centre for Gene Regulation & Expression, University of Dundee, Dundee, Scotland, UK; [b]Glencoe Software, Inc. Seattle, WA, USA

ABSTRACT

The Open Microscopy Environment (OME) has built and released Bio-Formats, a Java-based proprietary file format conversion tool and OMERO, an enterprise data management platform under open source licenses. In this report, we describe new versions of Bio-Formats and OMERO that are specifically designed to support large, multi-gigabyte or terabyte scale datasets that are routinely collected across most domains of biological and biomedical research. Bio-Formats reads image data directly from native proprietary formats, bypassing the need for conversion into a standard format. It implements the concept of a file set, a container that defines the contents of multi-dimensional data comprised of many files. OMERO uses Bio-Formats to read files natively, and provides a flexible access mechanism that supports several different storage and access strategies. These new capabilities of OMERO and Bio-Formats make them especially useful for use in imaging applications like digital pathology, high content screening and light sheet microscopy that create routinely large datasets that must be managed and analyzed.

Keywords: imaging, data management, database, HCS, digital pathology

1. INTRODUCTION

Complex heterogeneous datasets are a mainstay of modern biology. In imaging it is now routine to record several gigabytes (GBs) to terabytes (TBs) of raw data, and then process these data by deconvolution or other signal recovery methods, and then perform several analysis steps to convert the image data to quantitative values to express a statistically valid measurement. Raw and processed datasets inevitably include the binary pixel data and one or more sets of metadata that describe the original sample, the image acquisition process, and a processing and analysis workflow. Tools that store and manage these complex datasets are increasingly important parts of imaging workflows, as they ensure reproducibility, provenance, integrity and consistency.

Several projects have built systems to manage these complex sets of data. Most use modern server-client architectures, where a single or small number of data servers and/or databases are linked to a much larger number of client applications. The most common implementation uses web browser-based client applications, but clients built in almost any other modern architecture are possible. All these systems use a defined point of access, an Application Programming Interface (API) to access the server application and the data it holds. The API abstracts the complexity of the server, and data storage systems from the client application and usually enables remote access to the datasets held by the server. The reduction in complexity comes with a cost, as access to the underlying data is limited by the API. In many cases, accessing large TB-sized datasets also entails a performance penalty, as data is marshaled through the API and transmitted to the client application. Accessing large datasets directly on a filesystem often improves performance but exposes the application developer to the complexity of the underlying data.

Medical Imaging 2015: Image Processing, edited by Sébastien Ourselin, Martin A. Styner, Proc. of SPIE Vol. 9413, 941307 · © 2015 SPIE · CCC code: 1605-7422/15/$18 · doi: 10.1117/12.2086370

One compromise between these extremes is to convert the underlying data to a common format. However, as dataset sizes grow, this conversion becomes an inevitable barrier—duplicating large datasets slows throughput and imposes an unacceptable storage burden. Several imaging domains, e.g., high content screening[1], light sheet microscopy[2] and digital pathology[3] now routinely generate multi-GB or even TB-sized datasets on a daily basis, making conversion of raw data to a standardized file format (e.g., DICOM, OME-TIFF) impractical in production imaging labs and facilities. The only practical solution is to read data in its original form, ideally while maintaining the advantages of defined, remote access afforded by an API.

Since 2000, the Open Microscopy Environment (OME) has built open source software interfaces that provide access to large and/or complex image datasets[4]. OME's Java-based Bio-Formats library reads >130 proprietary image file formats, and converts them into a common model for access by third party software[5]. OME's OMERO is a server-client platform that uses Bio-Formats to read image data, stores all metadata in a relational database for querying and access control and provides a single application programming interface (API) that supports access from Java, C++, Python, Matlab and web clients[6]. Bio-Formats and OMERO are installed in thousands of sites worldwide and are the basis for several on-line image data repositories[4]. Here we describe the next step in the evolution of OMERO and Bio-Formats that enables access to original data, while delivering high performance, enterprise-scale APIs that allow integration of the wide diversity of image file formats with a broad set of image processing and analysis tools.

2. SYSTEM ARCHITECTURE

Environment and availability.

All the software is written in Java (at least Java 1.6 is required) or Python and licensed under the GPL v2 license..

The architectural principles of Bio-Formats and OMERO have been described previously[5, 6]. Bio-Formats combines low-level file I/O libraries, a file format detection and file readers for those file formats it supports. OMERO compromises a middleware server application that connects a relational database, an image data repository, search index and tabular data stores with an image rendering engine and the API. Client applications in Java, Python and a Python-based web gateway are available.

3. REQUIREMENTS AND RESULTS

Original file access

Previous versions of OMERO e.g. version 4, used an internal repository employing an optimised binary file format for storing pixel data. This proprietary format was implemented as a random access binary array to enable rapid access to image pixel data by the server application and subsequent rendering and transfer to client applications. This strategy implied duplication and transformation of incoming data, which is unacceptable in large volume production imaging.

In OMERO 5, all access to binary data had to be achieved through the original data format, removing the data duplication and the computational cost of data conversion and the resulting loss of information. This required a complete re-evaluation of the Bio-Formats data access library to substantially enhance its performance against many different types of file formats. This is particularly challenging for file formats that store individual image planes in a multidimensional dataset. Wherever possible Bio-Formats access mechanisms to these data formats was substantially improved so that throughput when reading these data could be maximised.

File sets

To deal with the diversity of file formats, we codified the concept of a "*file set*"— a group of files related to one or more multi dimensional images stored in a defined multi-file or multi-directory structure. The file set concept allows Bio-Formats to treat complex metadata and image data structures as a single entity and manipulate them as such. Applications that use Bio-Formats, e.g. OMERO, use the file set concept through an API, facilitating access and management of heterogeneous data.

One consequence of implementing this concept is that Bio-Formats is quite sensitive to the format of the file sets it tries to read. Changes in the file structure used by a proprietary vendor usually cannot be identified by Bio-Formats. Given the substantial use of Bio-Formats by the community, we are now receiving updates to file formats from several companies before they are released to the public. This goes some way to improving our ability keep Bio-Formats up-to-date.

Managed Repository

To enable OMERO to use and work with file sets, a new subsystem has been built, OMERO.fs' *Managed Repository*. OMERO 5 stores image files (or symbolic links to them, see below) in their raw form, as acquired from the imaging system. Data import amounts to either a copy of original files or links to them.

A configurable template parameter defines a systematic layout of directories into which imports are placed. For example, imports can be placed under a directory named after the user importing, followed by a timestamped directory. Beneath that, all the individual directories and files of the file set will be reproduced exactly.

This transparency not only prevents locking users into yet another proprietary system but even facilitates third-party workflows. Though OMERO assumes ownership of the files (or links) that are stored in the Managed Repository with write-access limited to the server itself, any other entity can be given read access based on a site policy.

In place import

Another requirement for OMERO 5 was the ability to contend with increasingly large image datasets that, once written, could not be copied to another location. The sheer volume of data acquired in large-scale HCS, DP and, LSFM imaging experiments means that, in practice, data can only be written once on the storage system available in most institutions. This is because there is insufficient storage capacity, network bandwidth or even time to copy these large datasets once they have been written. For this reason we implemented an "in–place import" function, where soft or hard symbolic links are used to connect the OMERO Managed Repository to the source of original data. These links can be created easily using a command-line option, whether from a script run either by a user or a regular cron job. In-place import also allows a single OMERO installation to be connected to several different logical or physical data sources within an institution's network. This substantially increases the flexibility of storing and accessing data in OMERO.

Permissions

OMERO includes functionality for controlling access to data using its Permissions system. Users are included in groups and the access and control policies between different users within a group and between groups can be defined. Specified datasets can be published to the outside world. This has proven to be a particularly attractive and popular capability of OMERO and is used by many institutions to share data in collaborations or publish data for the whole world-wide community to view and browse.

Similar access and control concepts occur in filesystems. Thus, in delivering OMERO 5, it was critical to rationalise filesystem-level permissions with those defined by the users of an OMERO installation. Critically, we were unable to design and build a system that allowed users freedom to change files at the filesystem level once they were imported by OMERO. To rationalise filesystem and OMERO permissions, the OMERO 5 Managed Repository can be accessed by users but with read-only permissions. Users add data through OMERO clients, which set the correct file paths and filesystem permissions. This insures that original image data files are available in the OMERO Managed Repository but can only be changed through access by the OMERO API. In the case of in-place import, the symbolic links stored in the OMERO file repository are marked for read-only access by all users except OMERO. These links connect to other filesystems with their own permissions structure that is outside the control of OMERO.

API Transparency

Making a fundamental change to the data access strategy for a data management application risks breaking the access methods for all existing client applications. To support the broad and growing community of OMERO developers, we committed to hold the OMERO API stable through the first year of OMERO 5 releases. This ensured that all developers working against the OMERO API could depend on a stable API, while taking advantage of the new data access and import strategy. In OMERO 5, the only changes in the OMERO API from version 4.4 were additions; this ensured that existing applications using legacy API components were unaffected. The consequence is that a completely new server application has been delivered with little or no effect to OME's or others' existing client applications running in production.

Flexible data access

As dataset sizes increase, so does the demand and requirement for larger computational resources to generate analytic results. Flexible access to data, to accommodate as many different computational resource configurations as possible is therefore important. As just one example, a large timelapse dataset might be processed by distributing each of many hundreds or thousands of time points across different nodes on a compute cluster. Each node will usually access data for processing through some kind of clustered filesystem e.g. GPFS, Lustre, etc. In this case, all processing nodes will access metadata through a central resource but will access binary image data directly through the clustered file system, the only requirement being a common view of the filesystem.

OMERO 5 enables this kind of flexibility. Each node uses a single OMERO installation as a metadata store to read necessary parameters and write analytic results, but accesses binary image data directly from the filesystem, bypassing the OMERO application. Calculated results can be stored via the OMERO API or stored on some external resource and linked via a URL. The detailed configuration of such a set up depends greatly on the particular configuration of cluster nodes, filesystem, etc. By not proscribing a single data access method, OMERO 5 provides substantial flexibility for defining large distributed compute resources, and supports the widest possible range of configurations.

Using Bio-Formats and OMERO 5 in HCS

High content screening data consist of images recorded at different physical positions in defined formats, usually either 96- or 384-well plates. This plate format is used to conduct large-scale assays with libraries of small molecules, siRNAs, or genome editing agents. Each well is imaged in turn, sometimes at several different locations, by an automated microscope. Images are then organized in successive logical containers of well, plate, and screens.

Bio-Formats and OMERO use the OME ScreenPlateWell Data Model[7] to read and store metadata related to an HCS dataset. The various commercially available HCS imaging systems all write data in proprietary file formats. Bio-Formats 5 reads these files directly without conversion, defining them as a single file set, and making them accessible to other applications, e.g. OMERO, by converting metadata into the OME ScreenPlateWell model. Figure 1 shows an XDCE HCS file set read with Bio-Formats 5 and visualized in OMERO 5. All plate metadata is read directly from the incoming file and stored in OMERO, while the images themselves are read directly from the XDCE format.

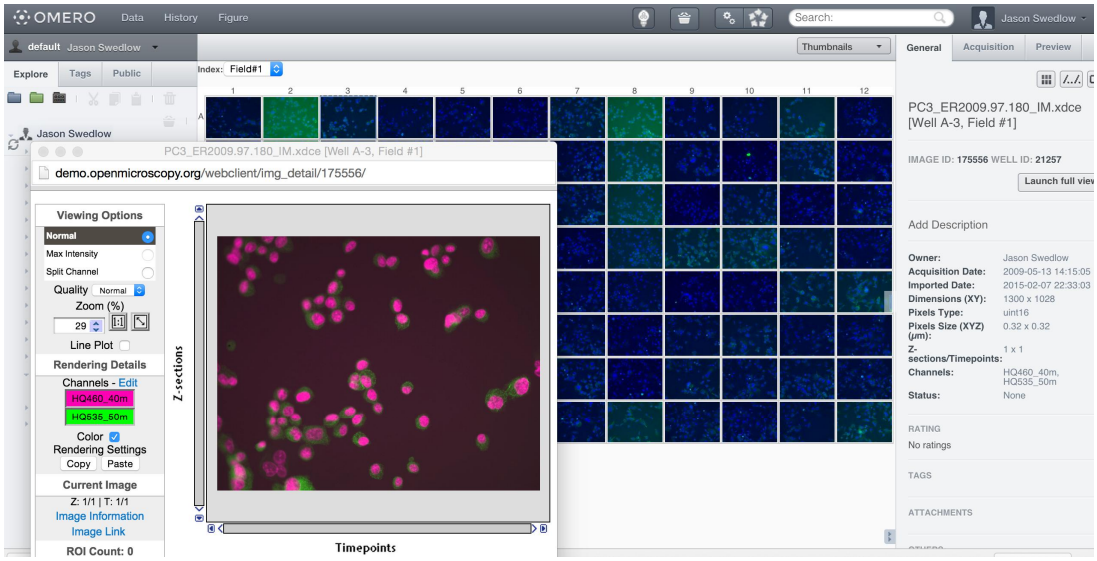

Figure 1. HCS plate image from an XDCE file. Each well consists of two fields, and two separate channels. All metadata is read by Bio-Formats and stored in OMERO for recall and display.

Using Bio-Formats and OMERO 5 for digital pathology

Whole slide images (WSIs) are commonly used for visualizing tissue sections mounted on slides for histopathological examination**[3]**. Imaging is performed by scanning slides, recording small strips or lines at magnifications used by pathologists (10x, 20x, 40x objective lens). Because tissue preparations are often on the order of a few mm in size, full images containing 10^8 – 10^9 pixels are quite common, with file sizes reaching several GBs. Although a DICOM specification for histopathology data exists [8], most commercial imaging systems store data in proprietary file formats. These include the full resolution image and a series of derived images sampled at reduced resolution, with a specified tiling at each resolution. Bio-Formats 5 reads these multi-resolution pyramids and their specified tiling natively, and thus enables direct access to these large files from an OMERO 5 system. As with HCS data, image metadata is stored in and accessed from the OMERO database.

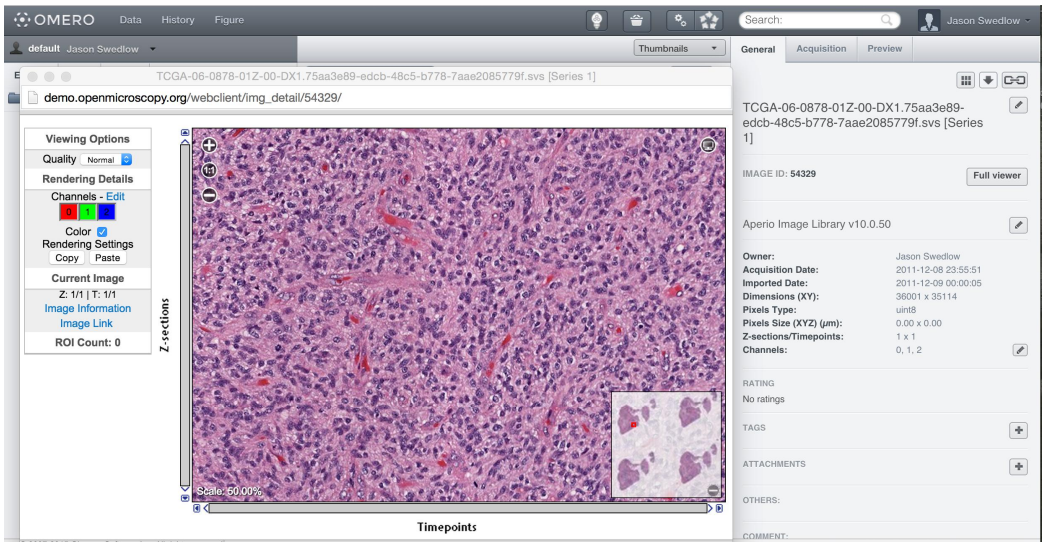

Figure 2. Tiled, multi-resolution image in SVS format of an H&E stained section of glioma, accessed and visualized with Bio-Formats 5 and OMERO 5. Image data from the Pathology Analytic Imaging Portal [9].

Using Bio-Formats and OMERO for Light Sheet Microscopy Data

Light sheet fluorescence microscopy (LSFM) is a new imaging technique that enables imaging of mm-sized embryos and tissues at diffraction-limited resolution[10]. The technique uses distinct fluorescence illumination and emission light paths, where the excitation light is introduced via a thin light sheet, and resulting emission recorded along an orthogonal axis. The result is a significant reduction in out of focus blur and substantially improved contrast. It is especially suited for imaging living tissues and developing embryos and thus is often used in timelapse mode, recording several hundred optical sections per timepoint. In many cases, several views at defined angles are recorded and then reconstructed to provide a single 3D image of the specimen. Recording several optical sections, multiple views and many timepoints means that LSFM datasets are often hundreds of GBs to several TBs in size, creating challenges for processing, visualizing and sharing these data. Figure 3 shows an LSFM dataset read by Bio-Formats 5 and stored in OMERO. The different views have been represented as different channels to aid visualization.

4. CONCLUSION

We have built Bio-Formats and OMERO 5 specifically for reading and handling the large GB- to TB-sized datasets that are becoming more common in biological and biomedical imaging. These open source tools support an array of large image datasets and lay the foundation for deploying large compute resources to process them.

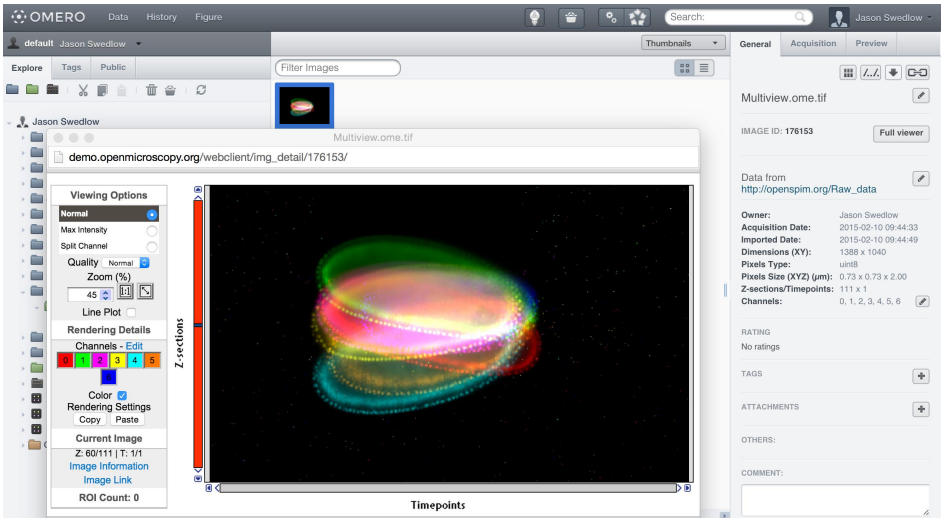

Figure 3. Multi-view LSFM data accessed and visualized with Bio-Formats and OMERO 5. The different angular views are stored as different channels to aid visualization. Data downloaded from OpenSPIM Wiki (http://openspim.org/Raw_data).

ACKNOWLEDGEMENTS

The authors thank all the Bio-Formats and OMERO user community for helpful feedback and suggestions for improvements to OME software. This work was supported by a Wellcome Trust Strategic Award (095931/Z/11/Z) and the BBSRC (BB/L024233/1).

REFERENCES

[1] O. J. Trask, Jr., A. Davies, and S. Haney, "High-content screening. Introduction," J Biomol Screen, 15(7), 719 (2010).

[2] P. J. Keller, A. D. Schmidt, J. Wittbrodt *et al.*, "Reconstruction of zebrafish early embryonic development by scanned light sheet microscopy," Science, 322(5904), 1065-9 (2008).

[3] L. Pantanowitz, "Digital images and the future of digital pathology," J Pathol Inform, 1, (2010).

[4] J. R. Swedlow, and K. W. Eliceiri, "Open source bioimage informatics for cell biology," Trends Cell Biol., 19(11), 656-660 (2009).

[5] M. Linkert, C. T. Rueden, C. Allan *et al.*, "Metadata matters: access to image data in the real world," J. Cell. Biol., 189(5), 777-82 (2010).

[6] C. Allan, J. M. Burel, J. Moore *et al.*, "OMERO: flexible, model-driven data management for experimental biology," Nature methods, 9(3), 245-53 (2012).

[7] J. R. Swedlow, C. Rueden, J.-M. Burel *et al.*, [High Content Screening, Science, Techniques, and Applications] Wiley, 317-328 (2008).

[8] R. Singh, L. Chubb, L. Pantanowitz *et al.*, "Standardization in digital pathology: Supplement 145 of the DICOM standards," J Pathol Inform, 2, 23 (2011).

[9] F. Wang, J. Kong, L. Cooper *et al.*, "A data model and database for high-resolution pathology analytical image informatics," J Pathol Inform, 2, 32 (2011).

[10] P. J. Keller, A. D. Schmidt, A. Santella *et al.*, "Fast, high-contrast imaging of animal development with scanned light sheet-based structured-illumination microscopy," Nat Methods, 7(8), 637-42 (2010).