OME User’s Meeting 2011

Kevin Eliceiri
(eliceiri@wisc.edu)

Curtis Rueden
(ctrueden@wisc.edu)

Laboratory for Optical and Computational Instrumentation (LOCI)
http://www.loci.wisc.edu
U. Wisconsin at Madison
Mission of LOCI:

- New optical instrumentation to facilitate studies of the dynamics of living specimens.
- Better software for capture and visualization of dynamic, 3-D biological events
- Been OME Development partner since 2003
- Image Informatics for multidimensional data
  - spatial and temporal
  - spectral and lifetime dimensions
- polarization
Our data:

Fluorescence, Spectra, Lifetime ($\lambda, \tau$)

Time-Lapse ($t$)

Dynamics

Chemistry

Physics

Phase, Polarization, Scatter, Harmonics ($\phi, \theta, ...$)

Structure

Space ($x, y, z$)

Supporting Technologies

Hardware
Acquisition software
Analysis and Visualization
Data Management
Overview of our OME efforts

• Specific OMER0 linked applications
  • BK Cho in Murphy lab on Omero.searcher
  • Forward Project for data dissemination

• Originated OME-TIFF in 2004
  • Now fully integrated into OMER0
  • Used by many open and commercial tools
  • Our current focus on robust tools to read and write OME-TIFF

• Originated Bio-Formats in 2005
  • Partnership with OME and Glencoe
  • Over 100 formats
  • Over 25,000 installations
  • Recent focus on native bindings

• XML Schema Improvements for Acquisition
  • Our WiscScan software and now MicroManager
  • Plans to extend to others that want richer “OME-TIFF”

• Interoperability between OME and other tools

• ImageJ 2.0 (ImageJDev.org)
Bio-Formats: the tool for interoperability
Open Source Toolkit Development

Interoperability

- EAR Sight
- OME
- Bio-formats
- ImageJ
- Fiji Is Just ImageJ
- CellProfiler
- µManager
LOCI/UW-Forward Collaboration

• Goal:
  – Provide access to LOCI datasets via Forward:
    • UW Libraries’ system-wide resource discovery application (forward.library.wisconsin.edu)

• Requirements:
  – Mapping OME-XML elements to METS/MODS elements
  – Adding description and opt-in functionality to WiscScan
  – Automating metadata transformations (OME to METS) and export of data/metadata to Forward
Getting Data into Forward

- Step 1: Add descriptive metadata in WiscScan
  - Title (Project Name)
  - Experiment description (Description)
  - Keywords (Annotations)
- Step 2: Opt in
  - Check-box in WiscScan experiment window
- Step 3: WiscScan and Forward do the rest
  - “Thumbnail” and metadata automatically uploaded to an OMERO directory that Forward harvests regularly
WiscScan Fields
\Needed for Forward

- Project Name
- Description
- Annotations

- Check if you want to upload images to UW Forward project.
Sample LOCI Record in UW-Forward
Big Picture

• LOCI data fully searchable and discoverable in UW-Forward
• Users searching for books/articles will now find datasets alongside traditional catalog results
• Greater exposure for data and emphasis on research data as a valuable campus resource
• Can be extended to Micromanager
Do the Math

• The potential size of a biological dataset has exploded:
  – A typical biological image is 512 x 512 pixels
  – There can be 30+ slices in an image stack
  – There can be 100+ timesteps in a time series
  – There can be 30+ channels at each pixel

• 512 x 512 x 30 x 100 x 30 x 4 bpp > 80 GB
Desired Features for visualizing a modern biological dataset:

1. View image stacks in 2D and 3D
2. Fast animation of these stacks
3. Roam through all dimensions in real time
4. Useful ways to sift through data’s non-spatial dimensions, to discover meaningful data quickly
5. Maintain a reasonable memory footprint even when the dataset is larger than the computer’s available memory
ImageJ

- ImageJ is an excellent 2D image viewer
  - Developed by Wayne Rasband at NIH
- Supports image stacks (3D), but only one plane at a time
  - Plugins such as Volume Viewer allow for 3D rendering
  - “WiscScan 4D Data Browser” plugin facilitates quick browsing of 4D datasets produced by WiscScan
    - [http://www.loci.wisc.edu/4d/](http://www.loci.wisc.edu/4d/)
  - Another option for multidimensional support is the Image5D plugin
- Very large datasets can be handled using the Virtual Stack Opener plugin with reasonable performance
- Overall, ImageJ is excellent for #1, #2 and #3, but not as developed for #4 and #5
ImageJDev: an NIH-funded project to produce the next generation of ImageJ

Partnership between several institutions:
- LOCI at UW-Madison
- MBL at Woods Hole
- Broad Institute of MIT and Harvard
- Fiji group (MPI-CBG, Uni/ETH Zurich, etc.)

See also: imagejdev.org/collaborators
Guiding Principles

- Preserve backwards compatibility
- Maintain good performance
- Support N-dimensional imaging
- Use common input and output for data
- Minimize complexity
  - Introduce dependencies only when benefits outweigh disadvantages
- Employ modern software development practices
Vision

- What is ImageJ's greatest strength?
Vision

- What is ImageJ's greatest strength?
  - It's *extensible* by writing plugins
Vision

- What is ImageJ's greatest strength?
  - It's *extensible* by writing plugins
- How can we expand on this potential?
Vision

- What is ImageJ's greatest strength?
  - It's **extensible** by writing plugins
- How can we expand on this potential?
  - Plugins as *modular* “building blocks”
Vision

● What is ImageJ's greatest strength?
  - It's extensible by writing plugins

● How can we expand on this potential?
  - Plugins as modular “building blocks”

● What does modularity gain us?
Vision

- What is ImageJ's greatest strength?
  - It's **extensible** by writing plugins
- How can we expand on this potential?
  - Plugins as **modular** “building blocks”
- What does modularity gain us?
  - Modularity facilitates **interoperability**
The Need

- Extensibility
- Modularity
- Interoperability
Aims

1. Improve ImageJ’s core architecture
   a) Separate data model from user interface
   b) Develop extensions framework for algorithms
   c) Broaden the image data model

2. Expand interoperability with other tools

3. Grow ImageJ community resources

See also: imagejdev.org/proposal
The Challenge

- How do we maintain compatibility?
  - Will plugins and macros still work?
  - Do other programs work with ImageJ 2.0?
Design

- Considered several design approaches
  - Iterative (modify existing IJ1)
  - Greenfield (new application)
  - Delegation (change IJ1's internals)
  - Adaptation (leave IJ1 alone)
- Adaptation: IJ2 includes IJ1 as a library
- IJ1 and IJ2 grow and evolve together
Progress

OUR COMPETITORS JUST MADE OUR NEW FIVE-YEAR PLAN MOOT.

WHILE WE WERE STRATEGIZING, THEY WERE DOING SOMETHING I BELIEVE THEY CALL "WORK."
Modular Design

- Divided software into functional components
- Facilitates a cleaner design
- Provides flexibility to those using ImageJ as a library
- Mid-April: 15K lines of code
- Mid-June: ~43K lines of code
  - Excludes ImgLib and IJ1
Progress: Compatibility

- Data structures converted between IJ1 and IJ2 behind the scenes
- Microscope icon indicates IJ1 legacy plugin
Progress: ImgLib

- ImgLib written by Stephan Preibisch & Stephan Saalfeld of MPI-CBG
- Many possible storage strategies
  - Data in array, file on disk, database...
- Type-independent algorithms and plugins
  - Signed and unsigned integer, floating point
  - Bit depths: 1, 8, 12, 16, 32, 64 bit
Progress: Bio-Formats

- Adapted ImageJ to use Bio-Formats natively for reading file formats
- Files are opened as N-dimensional, ImgLib-backed images
Progress: Declarative Plugins

- Existing plugin example:

```java
ImagePlus original = WindowManager.getCurrentImage();

GenericDialog gd = new GenericDialog("\"Tubeness\" Filter");
gd.addNumericField("Sigma: ",
    (calibration==null) ? 1f : minimumSeparation, 4);
gd.addMessage("(The default value for sigma “ +
    "is the minimum voxel separation.)");
gd.addCheckbox("Use calibration information", calibration!=null);

gd.showDialog();
if (gd.wasCanceled()) return;

double sigma = gd.getNextNumber();
boolean useCalibration = gd.getNextBoolean();

TubenessProcessor tp = new TubenessProcessor(sigma, useCalibration);
```
Progress: Declarative Plugins

- Declarative plugin example:

```java
@Parameter(label="Input image")
public ImagePlus original = null;

@Parameter(label="Sigma")
public double sigma = 1.0;

@Parameter(label="Use calibration")
public boolean useCalibration = false;

@Parameter(label="Output image", output=true)
public ImagePlus result = null;

public void run(String ignored) {
    if (original == null)
        original = WindowManager.getCurrentImage();
    TubenessProcessor tp = new TubenessProcessor(sigma, useCalibration);
    ...
}
```
Progress: Core Plugins

- Translating ImageJ core plugins to ImageJ2 and/or ImgLib—93 done so far
Progress: ROIs

- 2D planar ROIs only for now; more later
- Using JHotDraw (same as OMERO.insight)
Progress: CellProfiler

- CellProfiler is a tool for executing high-throughout image analysis pipelines
- Achieves better interoperability with ImageJ using the declarative plugin mechanism
Progress: Requirements

- Gathered feedback from the community
- Major areas of ImageJ
  - Data model & image processing
  - Visualization & user interface
  - Input & output
  - Segmentation & regions of interest
  - Scripting & plugins
Progress: Development Tools

- Web site
- Unit test suite
- Continuous integration: Hudson
- Source control: Subversion & Git
- Project management: Maven & Trac
Future Releases

- ImageJ v2.0.0-beta series due July-September
  - Focus on documentation and community feedback
- ImageJ v2.0.0-final is due on October 1
  - Finalized API for ImageJ v2.x
- For details, see:
  - http://dev.imagejdev.org/trac/imagej/roadmap
Benefits of ImageJ2

- What Will ImageJ 2.0 Do for Me?
  - Work with existing plugins and macros
  - Work with new, exciting plugins and scripts
  - Handle larger, more complex datasets
  - Multidimensional visualization tools
  - Easier to link with other software
  - Easier plugin management
ImageJ2 + OME

- OMERO is powerful server-side imaging software
- ImageJ is powerful client-side imaging software
- Both paradigms are valuable to scientists
- Many operations are common to client & server
  - E.g.: Bio-Formats is used for both
- Goal is to identify more areas for code sharing:
  - Big images (e.g., tiling with mipmaps)
  - Image rendering and thumbnails
  - Many others
Acknowledgements

- **Principal Investigators**
  - Kevin Eliceiri (LOCI), Rudolf Oldenbourg (MBL), Anne Carpenter (Broad)

- **Developers**
  - Grant Harris, Barry DeZonia, Aivar Grislis (ImageJDev)
  - Lee Kamentsky, Adam Fraser (CellProfiler)

- **Collaborators**
  - Wayne Rasband (ImageJ)
  - Pavel Tomancak, Johannes Schindelin, Albert Cardona (Fiji)
  - Stephan Preibisch, Stephan Saalfeld (ImgLib, Fiji)
  - Mark Longair, Jean-Yves Tinevez (Fiji)
  - Jason Swedlow, OMERO development team (OME)