doi:10.1017/S1431927616002300
Microsc. Microanal. 22 (Suppl 3), 2016

The Enabler Framework: an Object-Oriented Toolkit for Microscopy Data Analysis

Michael K. Kundmann

e-Metrikos, Pleasanton, USA

Present-day microscopy data sets comprise numerous detector readouts that densely sample specimen properties along multiple spatial dimensions and multiple signal modalities. Typical examples include STEM EDS/EELS spectrum images, TEM tomography image stacks, and multi-channel optical sections in confocal microscopy. Such rich and complex data call for novel software tools that enable microanalysts to flexibly view, identify, extract, and quantitatively analyze their constituent signals. In the past, microscopy software development focused on the mathematical data reduction and analysis algorithms needed to isolate and quantify features or signals in digitized images and spectra [1], including filters for feature enhancement and detection, cross-correlation for drift correction and data alignment, model fitting, and various types of principal component analysis. These have been deployed within programs that run the gamut from proprietary analysis software packaged with commercial instruments to open-source software developed for general microscopy data processing (e.g. Hyperspy project [2]). However, despite the many packages, accessibility can be limited (due to high cost or to steep learning curves) and applying the tools of one package to data obtained in another can be difficult (due to proprietary data formats, incompatible abstractions, or overly specific application focus).

The Enabler framework seeks to address the need for a flexible, capable, accessible, and unified software system for analyzing multi-dimensional, multi-modal microscopy data sets. It is designed as a complete object-oriented application development framework that allows for fully polished apps with approachable UIs for routine users, as well as code-level class and object access for re-use and extension by researchers and developers. The Enabler effort has so far focused on defining a generic microscopy data model and developing a modular architecture based on the controller concept. Controllers, first devised for programs with graphic UIs (MVC pattern) [3], act as command dispatchers and mediators that keep subsidiary model data and views (UIs) in sync. In the Enabler framework, generalized controllers serve as self-contained program modules responsible for particular data-analysis (or other application-specific) tasks. Each controller manages its data and parameters (including loading from and archiving to persistent storage), one or more task-relevant UIs, associated algorithms, and references to other (helper) controller objects. The inter-object references enable arbitrarily complex applications to be constructed as networks of collaborating controllers.

Enabler's generic microscopy data model follows findings of recent work towards a standard hyper-dimensional microscopy data exchange format [4]. In the HMSA proposal, a microscopy data set consists of two parts, a multi-dimensional binary array and a text-based, hierarchically tagged (XML) description of the array specifying how to load and physically interpret it. This makes the data structure transparent and allows it to be captured in non-proprietary formats, thereby improving archive longevity. It also permits efficient access to both the binary array data and its metadata, allowing frequent updates to the latter (which may contain calibrations and analytical results) while avoiding time-consuming re-saving of the measured data array (which tends to be large and generally should not be altered).
With the HMSA effort as guide, Enabler defines classes for a number of data abstractions, including: NumericBrick, PhysicalQuantity, PhysicalRange, PhysicalArea, and PhysicalDatum. NumericBrick encapsulates the multi-dimensional data array, providing functions for extracting data subsets (slices)

and performing math operations on the data (e.g. arithmetic, projections, integrations, and FFTs). PhysicalQuantity represents physically dimensioned parameters (e.g specimen feature size, lattice spacing, and differential cross-section) and their math functions. PhysicalRange and PhysicalArea extend the PhysicalQuantity concept to 1D and 2D selections on a data set. PhysicalDatum represents a complete, physically calibrated data set, with metadata and named ranges and areas of interest.

A key task for any data analysis package is resolving measured data into component signals. This often involves processing a finely sampled data array, such as an image, to extract high-level features from the pixel-level data. The human visual system exemplifies such an abstraction process as it picks apart a scene to detect and identify nearby physical objects (e.g. a coffee mug on a cluttered desk). Although this sort of data abstraction can be viewed as an algorithm that takes an image as input and returns a list of detected features, developing understanding of how the brain does this points to something more like a flexible network of interacting modules, rather than a processing pipeline. Enabler's controller architecture is well suited to experimentation with such data-abstraction object networks (DAO-Nets).

The self-archiving feature of Enabler controllers facilitates repetition, adjustment, and review of complex data analyses. In effect, an analysis goes from being a sequence of user-triggered function calls to a gradually compiled living document or archive, one that is automatically generated as analysis proceeds and that can immediately be applied to other similar data sets. This living archive fully captures all aspects of the analysis and can be readily shared for detailed peer review.

Since Enabler is rooted in high-level concepts and a generic microscopy data model, it can potentially be implemented within any object-oriented development environment. Early efforts explored possible implementations using the .NET (Windows) and Cocoa (MacOS) frameworks, but since they are not specifically designed for scientific applications, even a rudimentary Enabler implementation is challenging within either of them. Their use in rapidly evolving consumer technologies also make them fast-moving (i.e. unstable) development environments with which it is difficult to keep pace. Good progress towards a basic implementation of the Enabler framework has been made with the surprisingly capable and stable DigitalMicrograph scripting (dm-script) system [5, 6]. Early experiences and results obtained with this experimental version of the framework will be presented.

References:

[1] D.B. Williams and C.B. Carter, "Transmission EM", 2nd ed. (Springer, 2009) chs. 31, 35, and 39.
[2] F. de la Peña *et al*, Hyperspy 0.8, (15 Apr 2015), http://dx.doi.org/10.5281/zenodo.16850.
[3] E. Gamma *et al*., "Design Patterns", (Addison-Wesley, 1994) p. 4.
[4] M. Kundmann *et al*., Microscopy and Microanalysis, **17** Suppl. 2, (2011) p. 860.
[5] http://portal.tugraz.at/portal/page/portal/felmi/DM-Script
[6] http://www.gatan.com/products/tem-analysis/gatan-microscopy-suite-software