Design and implementations of the new HUPO Proteomics Standards Initiative's mass spectrometer output file standard format: *mzML 1.0*

Eric W Deutsch[1], Pierre-Alain Binz[2], Darren Kessner[3], Matt Chambers[4], Luisa Montecchi-Palazzi[5], Jim Shofstahl[6], Josh Tasman[1], Randall K Julian[7], Fredrik Levander[8], Puneet Souda[9], and Lennart Martens[5]

[1]Institute for Systems Biology , Seattle, WA; [2] Swiss Institute for Bioinformatics and Geneva Bioinformatics, Geneva, Switzerland; [3] Cedars-Sinai Medical Center, Los Angeles, CA; [4] Vanderbilt University, Nashville, TN, [5] European Bioinformatics Institute, Hinxton, UK, [6] Thermo Fisher, San Jose, CA, [7] Indigo Biosystems, Carmel, IN, [8] Lund University, Lund, Sweden, [9] University of California Los Angeles, Los Angeles, CA

Overview

mzML is a new data format for the storage and exchange of mass spectrometer output files. It follows on the successful mzXML and mzData formats. mzML has been designed by merging the best aspects of both previous formats into a single unified format that is intended to replace all earlier formats.

- Version 1.0.0 just released
- Accompanied by a controlled vocabulary and semantic validation rules
- Many implementations of the format already exists, insuring quick adoption of the format
- Developed with full participation of academic researchers, hardware and software vendors
- Vendors have committed to supporting the new format once released.
- Format has been tested with several instance documents and many implementations of the format during beta testing
- mzML is expected to replace mzXML and mzData, but not expected to completely replace vendor binary formats.

Natively mass spectrometers store output in a variety of proprietary formats
- Hinders data sharing
- Makes writing vendor-neutral software difficult

Solution: develop a vendor-neutral open format
- that all vendor software can write
- vendor-neutral software can read and write

- NativeID references back to the original scan references in the source data

mzML is a common open format to record the output of mass spectrometers prior to database searching or other downstream processing of the spectra. It is expected that by 2009:

- Instrument vendors will write out or convert to mzML
- Search engines or other spectrum processing software will read and process mzML
- Data repositories will accept, process, and store mzML documents

History

mzML has been under development for two years with full participation of academic researchers, hardware and software vendors. It was first conceived at the PSI meeting in 2006, two years after mzXML and mzData were released. The original working name of dataXML was changed to the final mzML name in Lyon. The format was submitted to the PSI document process in November 2007 wherein it passed through formal internal and then community review. Version 1.0.0 of the format was completed just prior to this conference.

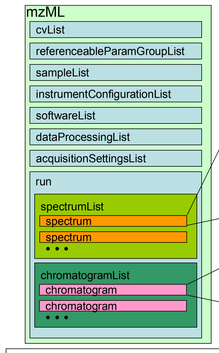
Early Development

Final Development

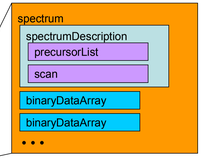
Maintenance on mzML will continue with the PSI Mass Spectrometry Standards Working Group. It is expected that the schema will remain stable, but minor updates to the controlled vocabulary and semantic validation rules may be necessary.

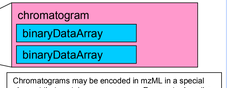
Schema Outline

The mzML schema is designed to contain all the information for a single MS run, including meta data about the spectra plus all the spectra themselves, either in centroided (peak list) or profile mode. The header at the top of the file encodes information about the source of the data as well as information about the sample, instrument and software that processed the data.



Each spectrum contains a header with scan information and optionally precursor information, followed by two or more base64-encoded binary data arrays.



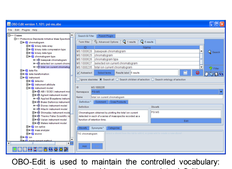


Chromatograms may be encoded in mzML in a special element that contains one or more cvParams to describe the type of chromatogram, followed by two base64-encoded binary data arrays.

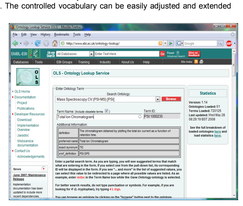
mzML may be enclosed in a special indexing wrapper schema to allow random access into the file, allowing software to pull out one or more arbitrary spectra.

Controlled Vocabulary

Much of the metadata encoded in the mzML is in the form of cvParams, an XML element that provides a reference to a specific concept within the PSI MS controlled vocabulary. Each term has an explicit and detailed definition, and may have information about its data type and what kind of units it requires, if any. The controlled vocabulary is edited in OBO format with the OBO-Edit software and is read in by most readers and writers of mzML. The controlled vocabulary can be easily adjusted and extended without modifying the schema.



OBO-Edit is used to maintain the controlled vocabulary; organize the structure, add new terms, update definitions.



The controlled vocabulary is easily parsed by software, such as reader and writer software, as well as central vocabulary services, like the Ontology Lookup Service web site.

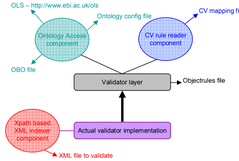
New term requests may be emailed to: psidev-ms-vocab@lists.sourceforge.net

Semantic Validator

One of the benefits of the previous mzData format was its considerable flexibility in allowing writers of the format to encode additional information relevant to the specific instrument or setup, even if it cannot be handled by all software. However, this considerable flexibility led to different dialects as the same information could be encoded in different ways. We have solved this problem with the semantic validator tool for mzML. The semantic validator can:

- Ensure that an mzML document is well formed and conforms to the xsd XML schema
- Ensure that controlled vocabulary terms are used in the correct places in the document
- Allow alternate rules based on the type of data being written
- Allow different levels of compliance (e.g., basic mzML, MIAPE-MS compliant mzML)
- Semantic rules can be updated along with the controlled vocabulary without changing the schema
- Available as a web page (see below) or as a standalone tool

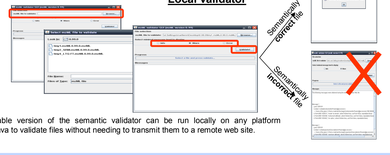
Semantic validator: concept



On-line validator

Online validator at the ProDaC site allows anyone to upload any file and perform semantic validation on any mzML file

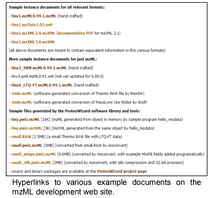
Local validator



A downloadable version of the semantic validator can be run locally on any platform supporting Java to validate files without needing to transmit them to a remote web site.

Example Instance Documents

In order to exercise the schema and demonstrate that the various use cases have been adequately modeled, we have developed several example instance documents. Some of the documents are hand-crafted with an ordinary editor, while others are written out as a software test as part of the ProteoWizard reference implementation. In addition, several instance documents are conversions of real data files using the reference or other implementations of converters.



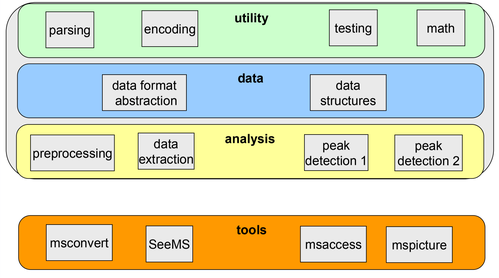
Hyperlinks to various example documents on the mzML development web site.

A sample snippet of an example mzML document, showing the top header portion of the file.

ProteoWizard

The ProteoWizard software project, initiated by the Spielberg Family Center for Applied Proteomics at the Cedars-Sinai Medical Center, provides a modular and extensible set of open-source, cross-platform tools and libraries. The tools perform proteomics data analyses; the libraries enable rapid tool creation by providing a robust, pluggable development framework that simplifies and unifies data file access, and performs standard chemistry and LCMS dataset computations. During the final stages of mzML development, refinement, and testing, the ProteoWizard library has provided the testing and reference implementation of mzML.

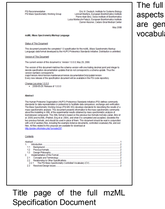
High level architecture of ProteoWizard



- C++ library, with modular design for testability and extensibility
- builds with native compilers on all major platforms (MSVC on Windows, gcc on Linux, XCode on OSX)
- open source license suitable for both academic and commercial projects (Apache v2)
- internal data model is a one-to-one translation of mzML data elements to C++ data structures
- plug-in Reader interface for reading of both open and vendor proprietary data formats: mzXML, Thermo RAW, MGF, with more Readers in development
- *msconvert* tool provides general file format conversion, including native centroiding and zlib compression
- *SeeMS* and *mspicture* visualization tools allows visualization of mass spec data
- used by RAMP and TPP for mzML support
- CLI binding allows use from .NET languages (C++/CLI; C#, VB.NET); SWIG bindings for scripting (from Java, Python, Perl, R) in development.
- an "Application Note" describing ProteoWizard has been accepted for publication by the journal Bioinformatics
- http://proteowizard.sourceforge.net – developer contributions are welcome!

Documentation

The full specification of the format is presented in a specification document that described various aspects of the format as well as all details of the format elements. HTML and PDF documentation are generated by programmatically combining 4 different components: xsd schema, controlled vocabulary, semantic validation mapping file, example documents.



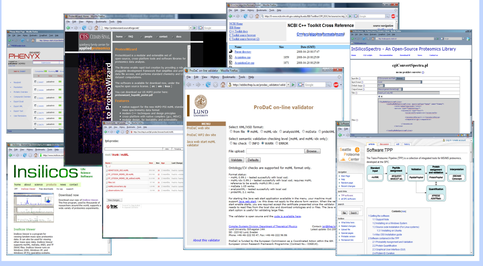
Title page of the full mzML Specification Document

Part of the HTML documentation page for mzML, showing the available documentation for the <sourceFile> element.

Software Implementations

The best way to test a new format is by implementing it in software. Inevitably as a format is implemented, one finds minor inconsistencies or missing features. The initial release of mzML is strengthened by the breadth of implementations that already exist and have exercised the various use cases:

- ProteoWizard C++ library reference implementation: reads mzML and writes mzML. Can convert mzXML and RAW to mzML.
- RAMP (Random Access Minimal Parser) C library can read mzML, mzXML, mzData files via same API. mzML reading performed with ProteoWizard
- Trans Proteomic Pipeline (TPP) can read and process data in mzML format via RAMP
- ISB format converters: ReAdW (Thermo), Wolf (Waters), mzWiff (ABI/MDS), Trapper (Agilent)
- Thermo Fisher beta RAW → mzML converter
- Semantic Validator and Java library reads and validates that a document is semantically correct
- Phenyx search engine can read and search spectra in mzML format
- NCBI C++ mzML reader classes
- Insilicos Viewer – file browser and spectrum viewer can read and display spectra from mzML, mzXML, mzData, RAW formats
- SeeMS file browser, spectrum viewer, chromatogram viewer, annotater for mzML, mzXML
- Proteios Software Environment includes converters for peak lists of various formats to mzML and performs reading of mzML files
- InSilicoSpectra open source library (Perl) has a spectrum file format conversion tool that reads mzML
- Hermes mzML ↔ mzData ↔ mzXML converter (Java)



Conclusions

The mzML format is now complete and mzML 1.0.0 is released. We encourage all authors and vendors to begin supporting this new format in new and updated software. The format includes the best features from pre-existing open formats and has additional support for chromatograms and some other features deemed highly desirable.

It is expected that the schema will remain stable for at least a year, hopefully more. However, the controlled vocabulary and semantic validation rules will continue to be updated and refined as all authors and vendors finish implementing their software for mzML.

To learn more, see the mzML Development Page:

http://psidev.info/index.php?q=node/257

The mzML effort has involved many people in the PSI and in the community. We gratefully acknowledge the contributions of:

Jari Hakkinen (Lund)
Brian Pratt (Insilicos)
Erik Nilsson (Insilicos)
Mike Coleman (Stowers)
Luis Mendoza (ISB)
David Shteynberg (ISB)
Lars Nilse (Manchester)
Benito Cañas (Madrid)
Lola Gutierrez (Madrid)
Alberto Medina (Madrid)
Trish Wheetzel (U Penn)
Eva Duchoslav (MDS Sciex)
Henning Hermjakob (EBI)
Angel Pizarro (U Penn)
Phil Jones (EBI)
Jimmy Eng (U Washington)
Kent Laursen (Indigo)
Sandra Orchard (EBI)
Chris Taylor (EBI)
Patrick Pedrioli (ETHZ)
Sean Seymour (ABI)
David Creasy (Matrix Science)
Howard Read (Waters)
Jim Langridge (Waters)
Jayson Falkner (U Michigan)
David Horn (Agilent)
Ruth McNally (Cardiff)
Ron Beavis (UBC)
Norman Paton (Manchester)
Ruedi Aebersold (ETHZ)
Marc Sturm (U Tuebingen)
Parag Mallick (CSHS)
Rune Philosof
David Sparkman (U Pacific)
Wilfred Tang (ABI)
Marius Kallhardt (Bruker)

PSI Steering Group
PSI Participants