**The ten minute guide to implementing mzIdentML**

Andy Jones, University of Liverpool, [andrew.jones@liv.ac.uk](mailto:andrew.jones@liv.ac.uk)

March 2012

**Introduction**

The purpose of this guide is to de-mystify mzIdentML and show that writing an exporter to mzIdentML is quite straightforward. If you would like help with implementing mzIdentML, please email me ([andrew.jones@liv.ac.uk](mailto:andrew.jones@liv.ac.uk)) or send a mail to the mailing list: [psidev-pi-dev@lists.sourceforge.net](mailto:psidev-pi-dev@lists.sourceforge.net). You can sign up for the mail list here: <https://lists.sourceforge.net/lists/listinfo/psidev-pi-dev>.

The stable version of mzIdentML is version 1.1, resources about it can be found here: <http://www.psidev.info/mzidentml/> and on our google code repository <http://code.google.com/p/psi-pi/>. An overview of the whole schema is provided in .

An example file showing all the basic features of mzIdentML accompanies this document, located here:

<http://code.google.com/p/psi-pi/source/browse/trunk/specification_document/specdoc1_1/TutorialExample/55merge_omssa_minimal.mzid>

Figure 1 Graphical overview of the mzIdentML file structure.

1. **Reporting peptide-spectrum matches**

All peptide-spectrum matches (PSMs) are captured within the *SpectrumIdentificationList*. The results from searching each individual spectrum are captured in *SpectrumIdentificationResult*. One single PSM is captured in *SpectrumIdentificationItem*, which has an attribute rank to demonstrate the ordering of hits for the spectrum (Figure 2). Other attributes include chargeState, calculatedMassToCharge and experimentalMassToCharge.

*SpectrumIdentificationItem* captures the scores associated with the identification, such as e-values or ion score, using controlled vocabulary parameters, sourced from the PSI-MS CV [1]. If you would like to add a new CV term, please request it via this page: <http://www.psidev.info/index.php?q=node/440>.

*SpectrumIdentificationItem* references the peptide that was identified in the *Peptide* element, under the *SequenceCollection*. The combination of the peptide sequence and modifications must be unique within the *SequenceCollection* (to reduce redundancy if the same peptide has been identified many times).

*SpectrumIdentificationItem* also references *PeptideEvidence* elements, which capture the mappings from the *Peptide* sequence to all the protein sequences, captured in *DBSequence*, it can be found in (i.e. the presence of the peptide sequence along, assuming no protein inference).



Figure 2 - Peptide-spectrum matches are captured in *SpectrumIdentificationItem* (SII). SII has references to the *Peptide* sequence and *PeptideEvidence*. *PeptideEvidence* is a one-to-many mapping from a Peptide sequence to proteins, captured in *DBSequence*.

1. **Reporting protein inference**

Following a protein inference step, results are captured in *ProteinDetectionList*. All results are grouped under *ProteinAmbiguityGroup* to represent each group of proteins where there is ambiguity in peptide to protein inference (Figure 3). A single protein identification is represented as *ProteinDetectionHypothesis*, along with any associated scores or statistical values, captured as CV terms. CV terms are available for flagging that one *ProteinDetectionHypothesis* is the group representative.



Figure 3 - Individual identified proteins are captured within ProteinDetectionHypothesis. Groups of proteins where there is ambiguity in peptide to protein inference are captured within ProteinAmbiguityGroup.

1. **Search metadata**

Metadata about the search should be captured for the software package used, the peptide identification step (*SpectrumIdentificationProtocol*) and the protein inference step (if used – in *ProteinDetectionProtocol*). Various parameters are requested including tolerances, enzyme, modifications and so on, sourced from the PSI-MS CV or Unimod [2]. The two protocols can both specify a threshold that was used for identifications, such as p<0.05. This allows the exporter to write out identifications that are above and below the threshold, using the passThreshold = “true|false” on *SpectrumIdentificationItem* and *ProteinDetectionHypothesis*. The associated example file shows the encoding of a search with trypsin, fixed modification Carbamidomethylation on C, variable modification Oxidation on methionine, parent tolerance 1.5Da, fragment tolerance 0.8Da, 1 missed cleavage, searched with OMSSA version 2.1.4.

1. **Validating your file**

For the file to be a “valid” mzIdentML file, it must pass two tests. First, it must validate against the XML Schema. A good editor, such as Oxygen XML editor, can do this check for you. Second, various CV terms must be used correctly, as documented by a mapping file (<http://code.google.com/p/psi-pi/source/browse/trunk/cv/mzIdentML-mapping_1.1.0.xml>), which formally represents rules that particular PSI-MS CV terms must be used at various places in the file. This ensures that common terminology is used to represent scores, modifications and so on. There is an online validator that can check this for you: <http://www-bs2.informatik.uni-tuebingen.de/services/OpenMS/analysisXML/index.php>

1. **File extension and compression**

We recommend that exported files have the original file extension “.mzid” and that where appropriate they are compressed using gzip, to “.mzid.gz”.

1. **More advanced features**

This tutorial covers only the most basic implementation of mzIdentML for conveying PSMs and protein inference results with basic search options. More advanced (optional) features include:

* The ability to include one or more *MassTable* elements, showing the amino acid masses used by the search engine.
* Arrays detailing the identified fragment ions identified by the search engine.
* Non-standard enzymes specified by a regular expression.
* Combinations of multiple enzymes used independently or in tandem.
* Results of peptide mass fingerprinting searches.
* Searches against translated databases of DNA or RNA sequences.
* The use of multiple search engines to create consensus identifications.
* The specification of matches to decoy databases.
* Searches with N14 and N15 containing samples.

These features are demonstrated within the example files, located here:

<http://code.google.com/p/psi-pi/source/browse/trunk/#trunk%2Fexamples%2F1_1examples>

Specific details about what is allowed should be verified within the formal specification document, although be warned, the document is fairly lengthy to read in one go.

<http://code.google.com/p/psi-pi/source/browse/trunk/specification_document/specdoc1_1/mzIdentML1.1.0.doc>

Finally, if you are considering implementing mzIdentML, please sign up for the (low-traffic) mail list and get in touch with us (see Introduction). We will also add a link to your software from our website, which may also help bring it to a wider audience:

<http://www.psidev.info/index.php?q=node/408>

**References**

1. PSI-MS CV <http://psidev.cvs.sourceforge.net/viewvc/psidev/psi/psi-ms/mzML/controlledVocabulary/psi-ms.obo>
2. Unimod: <http://www.unimod.org>