A COMMON REPRESENTATION , STANDARDIZATION, ANALYSIS FOR DE NOVO SEQUENCING RESULTS

A Thesis Submitted to
the Graduate School of Engineering and Sciences of
İzmir Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

in Computer Engineering

by
Savaş TAKAN

December 2012
İZMİR

We approve the thesis of **Savaş TAKAN**

Examining Committee Members:

Prof. Dr. Sıtkı AYTAÇ
Department of Computer Engineering, İzmir Institute of Technology

Assoc. Prof. Dr. Jens ALLMER
Department of Molecular Biology and Genetics, İzmir Institute of Technology

Prof. Dr. Talat YALÇIN
Department of Chemistry, İzmir Institute of Technology

Assist. Prof. Dr. Tolga AYAV
Department of Computer Engineering, İzmir Institute of Technology

Assist. Prof. Dr. Yalın BAŞTANLAR
Department of Computer Engineering, İzmir Institute of Technology

13 December 2012

Prof. Dr. Sıtkı AYTAÇ
Supervisor, Department of Computer Engineering, İzmir Institute of Technology

Assoc. Prof. Dr. Jens ALLMER
Co- Supervisor , Department of Molecular Biology and Genetics, İzmir Institute of Technology

Prof. Dr. Sıtkı AYTAÇ
Head of the Department of Computer Engineering

Prof. Dr. R. Tuğrul SENGER
Dean of the Graduate School of Engineering and Sciences

ACKNOWLEDGEMENTS

I would like to express my sincere and deepest gratitude to my supervisors Assoc. Prof. Dr. Jens ALLMER and Prof. Dr. Sıtkı AYTAÇ for their guidance, patience, understanding, motivation and excellent support during my MSc study and preparation of this thesis.

I want to express my thanks to Canan HAS and Şule YILMAZ for all kinds of support in bioinformatics.

I also would like to express my thankfulness to the friends with whom I have worked in the JLAB for their kindness and help.

Finally, I gratefully thank my sister Işıl TAKAN and my family for their excellent support, understanding, and encouragement.

ABSTRACT

A COMMON REPRESENTATION, STANDARDIZATION, ANALYSIS FOR DE NOVO SEQUENCING RESULTS

Proteomics is the study of the proteins that can be derived from a genome. For the identification and sequencing of proteins mass spectrometry has become the tool of choice. Within mass spectrometry-based proteomics proteins can be identified or sequenced by either database search or de novo sequencing. Both methods have certain advantages and drawbacks but in the long run we envision de novo sequencing to become the predominant tool. Currently, there is no a general solution to store and share *de novo* sequencing results which diminishes the usefulness of these results. Hence, they need to be integrated and further analyzed and cannot be used directly as evidence for peptide identification and protein sequencing at present. In order to make improvements the field of *de novo* sequencing a standard is vitally important.

In an attempt to overcome the standardization problem, the *de novo* markup language (DNML) and *De Novo* MS Ontology (DNMSO) are developed. These standards provide many-to-many relationships between spectra and predictions, exchange and merging functions, showing all results, PTMs, compact, no using proprietary formats. Next, a programming interface is developed since it is thought that the missing of proper APIs as another obstacle, introducing a needlessly high learning curve for developers. It is standard, compact, modular, easily extensible and also have read, write, create, convert, supports current standards facilities. In order to allow the experimental proteomics community to analyze data stored in the DNMSO standard, Graphical User Interface is developed , DNMSO Analyzer, to provide some facilities such as reading of various spectra file formats, reading, viewing, summarizing of DNMSO, and several conversions from existing de novo results to DNMSO in DNMSO Analyzer.

ÖZET

DE NOVO SIRALAMA SONUÇLARININ GENEL GÖSTERİMİ, STANDARTLAŞTIRILMASI VE ANALİZİ

Proteomiks genomdan gelen proteinleri inceleyen bir bilim dalıdır. Günümüzde, kütle spektrometresi proteomiks alanında proteinleri tanımlamada ve sıralamada en çok tercih edilen araç konumundadır. Kütle spektrometresi kullanarak veri tabanı arama yöntemi ve de novo sıralama algoritmalarıyla proteinler tanımlanıp sıralanabilmektedir. Veri tabanı arama ve de novo sıralama yöntemlerinin belli avantajları ve dezavantajları bulunmaktadır. Buna rağmen, de novo sıralama yönteminin gelecekte diğer yönteme göre tercih edileceği düşünülmektedir. De novo sıralama yöntemi düşünüldüğünde, sonuçların kayıt etmek ve paylaşmak noktasında genel bir çözüm bulunmaması bu sonuçların etkili kullanımını düşürmektedir. BU sonuçların entegrasyonu ve analizi gerekmektedir fakat şu andaki koşullarda, bu sonuçlar peptid tanımlama ve protein sıralama noktasında bir kanıt olarak kullanılması mümkün değildir. Bütün bu problemleri çözmek için de novo sıralama sonuçlarının standartlaştırması noktasında önemli bir gereksinim bulunmaktadır.

Bu gereksimleri karşılamak için de novo markup language (DNML) ve de novo ms Ontology (DNMSO) standartları geliştirilmiştir. Bu standartlar spectra ve tahminler arasında çok-çok ilişki, değişim ve birleştirme fonksiyonları, bütün sonuçların gösterilmesi, PTM'leri içinde barındırması, öz oluşuyla, açık format sunmasıyla de novo sıralama sonuçlarının entegrasyonu ve analizinde kolaylık sağlayacağı düşünülmüştür. Bunun dışında, bu standarttın rahat kullanılışını sağlamak amacıyla alanda eksikliği duyulan programlama ara yüzü geliştirilmiştir. Öz, modüler ve kolaylıkla genişletilebilir özellikleri yanı sıra okuma, yazma, yaratma, çevirme, bu alandaki diğer standartları destekleme özellikleri ile bu alandaki eksiklik giderilmeye çalışılmıştır. Son olarak, analizleri kolaylaştırmak amacıyla grafik kullanıcı ara yüzü geliştirilmiştir. Ara yüz ile okuma, görüntüleme, özetleme, çevirme sağlanmıştır.

TABLE OF CONTENTS

LIST OF FIGURES v

LIST OF TABLES x

LIST OF ABBREVIATIONS xi

CHAPTER 1. INTRODUCTION .....

1.1. Proteomics 1

1.2. Mass Spectrometry 2

1.3. Computational Mehods 5

1.3.1. Database Search 6

1.3.2. De Novo Sequencing 8

1.4. Standardization 11

1.5. The Aim of the Study 13

CHAPTER 2. MATERIALS AND METHODS .....

2.1. Ontology 15

2.2. XML 15

2.3. Programing Language 16

2.4. Other Technologies 16

2.5. Test Data 17

CHAPTER 3. DNML: DE NOVO MARKUP LANGUAGE .....

3.1. The Standard and Model 18

3.1.1. XML Schema 18

3.2. The Implementation 22

3.3. Comparison 27

CHAPTER 4. DNMSO: DE NOVO MS ONTOLOGY .....

4.1. The Standard and Model 25

4.1.1. Ontology..... 26
4.2. The Implementation..... 33
4.3. Compression..... 35

CHAPTER 5. VISUALZATION AND ANALYSIS..... 36

CHAPTER 6. RESULT AND DISCUSSION..... 43

CHAPTER 7. CONCULISION..... 46

REFERENCES…… ..…….. 48

LIST OF FIGURES

Figure **Page**

Figure 1.1. General process of proteomics is shown. Sample preparation is the first step in proteomics. In this step, proteins are extracted from cells. In the second step, electrophoresis methods are used to separate proteins. Then proteins are cleaved into peptides since peptides are easier to detect in mass spectrometry. In the fourth step, mass spectrometry is used to detect peptides and peptides fragments. Finally, the sequence of the protein can then be determined computationally. 1

Figure 1.2. Overview over mass spectrometry. Firstly, a sample is loaded into the mass spectrometer, and undergoes vaporization. Secondly, the components of the sample are ionized by one of a variety of methods. Thirdly, the ions are separated according to their mass-to-charge ratio in an analyzer by electromagnetic fields. Next, the ions are detected, usually by a quantitative method. Finally, the ion signal is processed into mass spectra..... 3

Figure 1.3. In this figure one-stage mass spectrometry and tandem mass spectrometry is compared. In one-stage mass spectrometry, ions are separated in the mass analyzer before they reach the detector. In tandem mass spectrometry, a specific ion is selected in the first mass analyzer (MS1) and then subjected to collision in the collision cell, whereas the second analyzer (MS2) separates the product ions before they reach the detector..... 4

Figure 1.4. Identification methods are shown that database search methods are employed to identify proteins from complex mixtures. However, often databases are not available or despite their availability some sequences are not readily found therein. To overcome this problem *de novo* sequencing can be used to directly assign a peptide sequence to an MS/MS spectrum.. 6

Figure 1.5. Data search methods such as Sequest are shown. In the Sequest algorithm, a signal- processing technique called autocorrelation is

used to mathematically determine the overlap between a theoretical spectrum that has been derived from every sequence in the database and the experimental spectrum in question.The overlap is given in the form of a score, and the score to the next best matching peptide sequence is also often given. in the Mascot, it involves calculating the m/z theoretically predicted fragments for all the peptides in the database, and it is called probability-based matching.The predicted fragments are matched to the experimental fragments in a top-down fashion. 7

Figure 1.6. A peptide can fragment at any chemical bond in the molecule. Some frequently observed ions are named within the figure. The names of the N-terminal ions are indicated on the left of the respective fragmentation border whereas the C-terminal portions are named on the right. A subscript indicates their position within the sequence in respect to their mass. ifi: Internal fragment ion; scl: Side chain loss..... 9

Figure 1.7. Spectrum graph approach is shown to peptide identification. MS/MS spectrum is converted into spectrum graph. Each node represents m/z of peak with score s.Edges connect nodes if mass difference of nodes corresponds to an amino acid 10

Figure 3.1. Depicts the DNML root element and the contained Spectra, Predictions, and Modifications elements. The XML tree can be further expanded to reveal that Predictions, for example, contain Prediction elements and a Software element. 19

Figure 3.2. Shows how the raw data source, the spectrum, is modeled. Spectrum has attributes such as id, precursor m/z, precursor intensity. The peaks in the MS/MS spectrum can either be provided as a link to a file containing spectra in mzML or mzXML standard, or as contained data.. 19

Figure 3.3. The Prediction element contains Sources, which defines the spectra that led to the prediction and Sequence, which contains sequence and confidence attributes.. 20

Figure 3.4. Software represents the tool that made one or multiple *de novo* predictions. Publication element is used to add publications of that software.. 21
Figure 3.5. A post translational modification (PTM) is modeled as a change to an amino acid and a file with all currently known PTMs is provided to simplify handling of PTMs. Terminal modifications for peptides are also possible. 21
Figure 3.6. The DNML library has four packages. The façade package handles in-put and sends it to the controller. The controller finds, creates, and executes commands. The service package handles do-main tasks. 23
Figure 4.1. sample ontology element is shown in obo format. Each element has unque id as standard.. 26
Figure 4.2. Depicts the DNMSO root element and the contained Spectra, Predictions, and Modifications elements. The Ontology can be further expanded to reveal that Predictions, for example, contain Prediction elements and a Software element.. 27
Figure 4.3. shows how the spectrum is modeled. Spectrum has attributes such as scan id, filename, precursor m/z, precursor intensity. The peaks in the MS/MS spectrum can either be provided as a link to a file containing spectra in mzML or mzXML standard, or as contained data. 28
Figure 4.4. The Predictions element contains two elements. The Software element is useful to group predictions from different software tools. It is of note in this context that Predictions may thus be contained in Source multiple times. 29
Figure 4.5. Software represents the tool that made one or multiple *de novo* predictions. Publication element is used to add publications of that software since trouble sometimes are experienced when searching for the first publication of an algorithm... 30
Figure 4.6. The Prediction element contains sources which define which spectra led to the prediction, Sequence which contains sequence and confidence attributes, Score which has name and value

parameters and different score parameter in prediction algorithms are represented... 31

Figure 4.7. The proper modeling of post translational modifications PTMs is important today but will increase in importance in the near future when more algorithms will become able to predict unexpected PTMs. It is allowed user specified PTMs but also provide a list of about 100 known modifications. A post translational modification (PTM) is modeled as a change to an amino acid and a file with all currently known PTMs is provided to simplify handling of PTMs. Terminal modifications for peptides are also possible.. 32

Figure 5.1. In this figure, it is seen that there is tree view for summarized representation about opening files. Opening file has many predictions and the third prediction is expanded to see information about sources which have one source and sequence which show amino acid and confidences. 37

Figure 5.2. Figure show an overview of results section. It summarizes all information about opening files such as id which represent number of spectrum in table, Spectra which show spectra data files, size of prediction which represent that number of prediction is done in this spectrum file, Sequence which describe highest finding sequence in predictions of this spectrum file, confidence which represent confidence of highest finding sequence in predictions of this spectrum file.. 38

Figure 5.3. Software part shows all setting of it in clicked predictions. For example, software setting about pepNovo+ is seen in there.. 39

Figure 5.4. Prediction result is shown. In there, charts, sources and assumed charge about spectra of this prediction, finding sequence, confidence, confidence, calculated mass and scores of this sequence. What is more, there is a table which show sequence elements and confidence of these elements... 40

Figure 5.5. The Sequence is seen in this figure. In there, all spectra of selected prediction are represented by m/z and intensity values. What is more, all support elements is shown in each spectrum as

red. In addition, fragment tolerances of support elements are represented with green... 41

Figure 5.6. Each sequence element is shown in this figure. In this picture, one gap, amino acid and modified amino acid are represented. When clicking these elements, spectrum with m/z and intensity values, fragment tolerance and support are shown in chart... 42

LIST OF TABLES

Table **Page**

Table 1.1. Data file formats such as MzWiff, Raw, mzXML, mzData, MzML is shown in Table. 5

LIST OF ABBREVIATIONS

M/Z	**:** Mass/Charge ratio
MS	**:** A single-stage mass spectrometry
MS/MS	**:** tandem mass spectrometry
XML	**:** Extensible Markup Language
DNML	**:** The *de novo* markup language
DNMSO	**:** *De Novo* MS Ontology
CV	**:** controlled vocabulary
W3C	**:** World Wide Web Consortium
RDF	**:** Resource Description Framework
DOM	**:** The document object model
SAX	**:** The simple API for XML

CHAPTER 1

INTRODUCTION

1.1. Proteomics

Proteins are biochemical machines responsible for life. They copy, read and organize the genetic code stored in DNA, digest nutrients, attack pathogens, and direct growth. Signals coming from proteins enable cells to interact with each other and structural proteins hold organisms together (Figeys 2005).

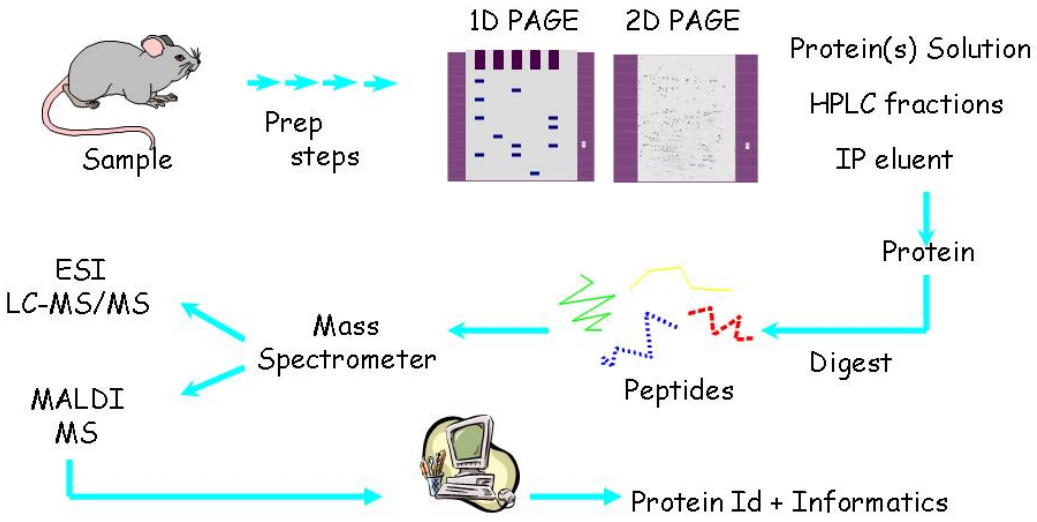

Figure 1.1. General process of proteomics is shown. Sample preparation is the first step in proteomics. In this step, proteins are extracted from cells. In the second step, electrophoresis methods are used to separate proteins. Then proteins are cleaved into peptides since peptides are easier to detect in mass spectrometry. In the fourth step, mass spectrometry is used to detect peptides and peptides fragments. Finally, the sequence of the protein can then be determined computationally (Source: proteomics.arizona.edu 2012).

In brief, proteomics is the study of proteins that can be derived from a genome. The word “proteome” comes from proteins (produced by an organism) represented by genome which is the whole set of genes (Liu and Hsu 2005). Its aim is to describe structure, function, and control of biological systems in health and disease (Aebersold and Mann 2003). General process is shown in Figure 1.1.

Proteomics can be categorized into three groups, 1) identification of proteins and their post-translational modifications that change proteins any time after it is translated, 2) finding of protein levels with potential application in a wide range of diseases, and 3) protein interactions using techniques such as mass spectrometry (Patterson and Aebersold 2003).

Recently, tandem mass spectrometry (MS) has become the main technology in providing many of the advances in the field of proteomics (Baldwin 2004). MS identifies, quantifies thousands of proteins among complex samples and allows for its application to proteins, peptides, carbohydrates, DNA, drugs, and many other biologically relevant molecules. A mass spectrometer creates charged particles from molecules, analyzes those ions to obtain information about the molecular weight of a compound (Domon and Aebersold 2006).

1.2. Mass Spectrometry

For the identification and sequencing of proteins, mass spectrometry has become the tool of choice in proteomics. In seconds, a tandem mass spectrometer is capable of ionizing a mixture of peptides with different sequences and measuring their respective parent mass/charge ratio (m/z), selectively fragmenting each peptide into pieces and measuring the mass/charge ratios of the fragment ions (MS/MS spectra of peptides). The peptide sequencing problem is then to derive the sequence of the peptides given their MS/MS spectra (Dancík et al. 2006).

Any mass spectrometer has three main components; the ion source, the mass analyzer, and the detector (Aebersold and Mann 2003) (Figure 1.2). The source yields ions from the biological sample, the mass analyzer selects ions of a certain m/z (in a mass-to-charge (m/z) ratio dependent manner), and the detector discovers the ions resolved by the mass analyzer (R Aebersold and Goodlett 2001).

The mass of a molecule is obtained by measuring the mass-to-charge ratio (m/z) with a mass spectrometer. Ions are generated by inducing either the loss or gain of a charge from a neutral species. The loss or gain of a charge is induced to generate ions. After that, ions are channeled into a mass analyzer which separates them in accord with m/z. The result is a spectrum which shows m/z and intensity (Downard 2004). The general process is shown in Figure 1.2

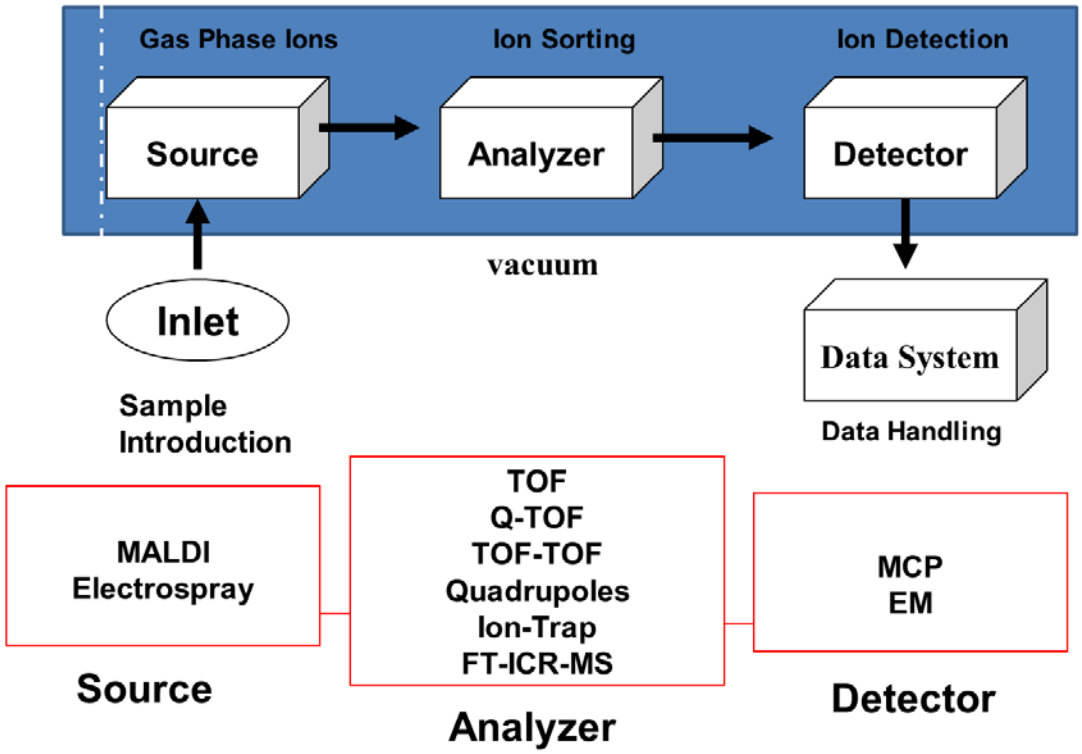

Figure 1.2. Overview over mass spectrometry. Firstly, a sample is loaded into the mass spectrometer, and undergoes vaporization. Secondly, the components of the sample are ionized by one of a variety of methods. Thirdly, the ions are separated according to their mass-to-charge ratio in an analyzer by electromagnetic fields. Next, the ions are detected, usually by a quantitative method. Finally, the ion signal is processed into mass spectra.

Mass spectrometry is used with several stages to get information about proteins. A single-stage mass spectrometry (MS) enables to measurement of peptide fragments in a protein sample and forms a “peptide mass fingerprint (PMF)” (Watson and Sparkman 2007) (Figure 1.3). With tandem mass spectrometry (MS/MS), two analyzers are used. Particular ions from a single stage are selected and then fragmented by collisions with an inert gas in a collision cell (Figure 1.3). A fragmented peptide is called a “precursor ion” or a “parent ion” and the resultant ions measured in the sequential analyzers are named as “product ions” or “daughter ions”. If a fragment is without any charges, it is cannot be detected directly. Hence, to improve identification, especially for complex sample mixtures, further fragmentation sections are performed in additional stages and this method is called “multistage mass spectrometry (MS^n)”(R Aebersold and Goodlett 2001).

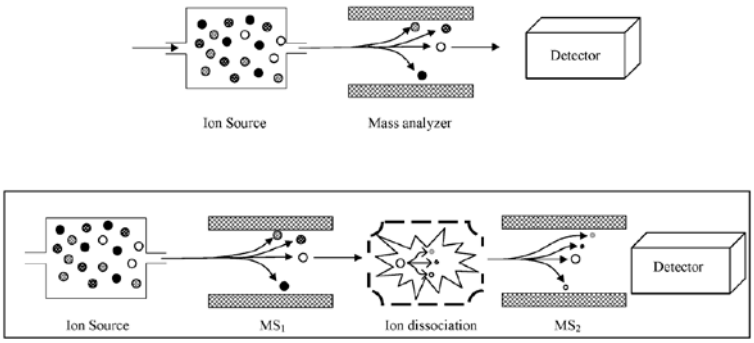

Figure 1.3. In this figure one-stage mass spectrometry and tandem mass spectrometry is compared. In one-stage mass spectrometry, ions are separated in the mass analyzer before they reach the detector. In tandem mass spectrometry, a specific ion is selected in the first mass analyzer (MS1) and then subjected to collision in the collision cell, whereas the second analyzer (MS2) separates the product ions before they reach the detector (Source: El-Aneed et al. 2009).

The large volume of data, produced in a typical mass spectrometry experiment, requires that computers should be used for data storage and processing. Until now, various proprietary data formats have been developed by different manufacturers of mass spectrometers in order to handle such data which makes it difficult for academic scientists to directly manipulate. To solve this limitation, several open, commercial XML-based data formats have recently been developed (Deutsch 2010).

1.2.1. Data File Formats

MzWiff is ABI/Sciex (QSTAR and QTRAP instrument) file format. Raw is Thermo Xcalibur file format, Micromass MassLynx directory format, PerkinElmer TurboMass file format. mzWiff and raw are commercial. mzXML (Pedrioli et al. 2004), an open data format, is developed by SPC/Institute for Systems Biology in order to store and exchange mass spectroscopy data. It is standard for ms/ms proteomics data. It is also the foundation of our proteomic pipelines. Raw, proprietary file formats from most vendors can be converted to the open mzXML format.This format has several versions such as 1.0 (called "msXML"), 2.0, 2.1, 3.0, and 3.1. 3.1 is the current version. mzXML schema is available for The official XML Schema components. mzData (Orchard et al. 2004) is the common file format and developed by The Human Proteome Organization (HUPO). It has similar functionality to mzXML. The Mass Spectrometry Standards

Working Group of the HUPO Proteomic Standards Initiative (HUPO-PSI) develop mzML (Martens et al. 2011) to represent the mass spectrometric data format. Version 1.1 was released in June 2009. Research centers and instrument vendors worldwide work collaboratively to replace mzXML and mzData formats with mzML.

Table 1.1. Data file formats such as MzWiff, Raw, mzXML, mzData, MzML is shown in Table.

Name	Domain	Type	Vendor	Version	Type
MzWiff	ABI/Sciex	commercial			
Raw	Xcalibur	commercial			
mzXML	ms/ms proteomics	Open	SPC/Institute	1.0 , 2.0, 2.1, 3.0, and 3.1.	XML
mzData	ms/ms proteomics	Open	HUPO	1.05	XML
mzML	ms/ms proteomics	Open	HUPO	1.1	XML

Parsing these data is important issue before analyzing. After that, Computation methods are needed to get information from these formats. The emphasis here is on the use of computers because most of the tasks in proteomics data analysis are highly repetitive or mathematically complex. The use of computers is absolutely indispensable in mining proteomics for information gathering and knowledge building (Xiong 2006).

1.3. Computational Methods

Mass spectrometry based proteomics enable to analyze, identify complex mixtures of proteins from various biological samples. A variety of methods may be employed to try to identify the proteins in the sample which are injected into the mass spectrometry instrument with the output of an MS or MS/MS run in hand. MS/MS database searching which search databases of theoretical MS/MS spectra, *de novo* sequencing investigate distances that correspond to the mass of a single amino acid (most amino acids have distinctly sized masses), and chaining these together to form a

partial sequence (Aebersold and Mann 2003). The general process is shown in Figure 1.4

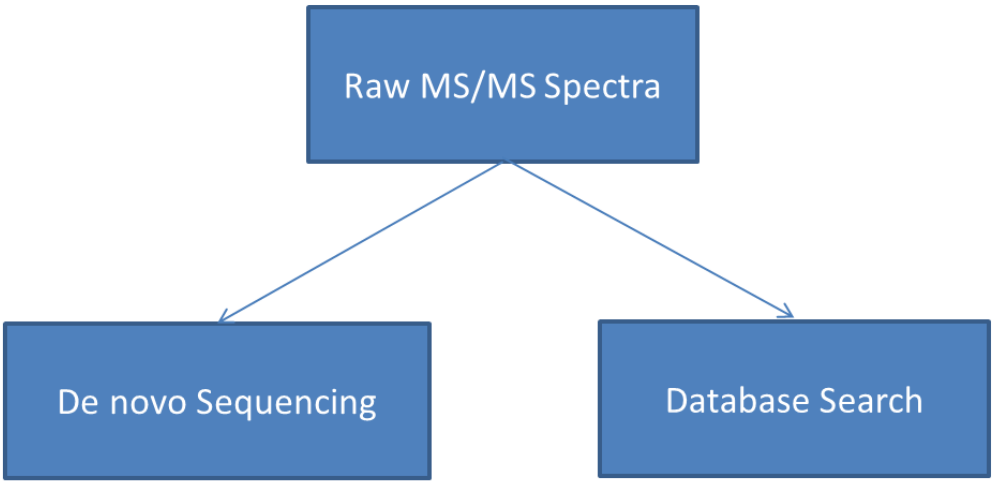

Figure 1.4. In this figure, identification methods are shown that database search methods are employed to identify proteins from complex mixtures. However, often databases are not available or despite their availability some sequences are not readily found therein. To overcome this problem *de novo* sequencing can be used to directly assign a peptide sequence to an MS/MS spectrum.

1.3.1. Database Search

One approach to determining the sequences of peptides is to use MS/MS data to search databases of synthetic peptide digests. Programs such as Sequest (Eng et al. 1994) and Mascot(Hirosawa et al. 1993) (Figure 1.6) are two widely used commercial tools which employ this approach. Sequest's approach (Figure 1.6) uses two pieces of information: the m/z ratio of the peptide before fragmentation (obtained from the first mass spectrometry step) and the MS/MS spectrum for generating identifications. Sequest investigate the m/z value of each peptide being analyzed in a master list of peptides generated from a computationally digested protein database, as in peptide mass fingerprinting (Steen and Mann 2004). Unlike PMF (peptide mass fingerprinting), the peptide's identity is determined by Sequest's approach using comparing the theoretical MS/MS spectrum of selected peptide in the list with the observed MS/MS spectrum. The theoretical MS/MS spectra are created by Sequest out of these peptides with a

model of how peptides fragment in the CID process (Seidler et al. 2010). A cross-correlation score (XCorr which is used to select the best match) is assigned to each theoretical peptide(Eng et al. 2011).

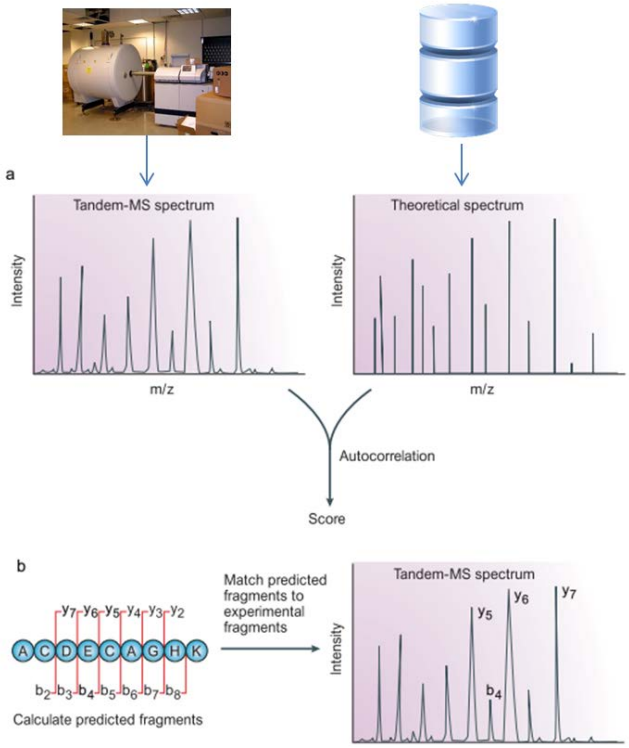

Figure 1.5. Data search methods such as Sequest are shown. In the Sequest algorithm, a signal- processing technique called autocorrelation is used to mathematically determine the overlap between a theoretical spectrum that has been derived from every sequence in the database and the experimental spectrum in question.The overlap is given in the form of a score, and the score to the next best matching peptide sequence is also often given. in the Mascot, it involves calculating the m/z theoretically predicted fragments for all the peptides in the database, and it is called probability-based matching.The predicted fragments are matched to the experimental fragments in a top-down fashion (Source: Steen and Mann 2004).

Though an automated high-throughput method for peptide identification is provided by these algorithms, a complete solution to this problem is not given by the current database search techniques. It is quietly presumed that sequence from the genome is correct. What is more, annotations of all protein coding genes are done. Because of many alternatively spliced genes, the latter condition, most of which are not adequately represented in the existing databases, is hardly ever met. Additionally, the identification of some peptide-spectrum matches is missed by database search

algorithms because they use the relatively simple scoring methods which have limitations. Another problem is that matches are missed. Hence, database search algorithms don't consider mutations/ polymorphisms or the presence of modified amino acids in the peptide (Frank and Pevzner 2005).

Data File Formats

Two standards have been implemented to store database search results in proteomics. An example for community standards is pepXML (Keller et al. 2005) that is an open data format developed at the SPC/Institute for Systems biology for the exchange ,storage, and processing of peptide sequence assignments of MS/MS scans. Another open data format is mzIdentML (Jones et al. 2012) that is published by The Proteomics Standards Initiative that released mzIdentML standard format for peptide and protein identification, which can represent the output of the most used search engines such as Mascot, The Open Mass Spectrometry Search Algorithm (OMSSA) , and X!Tandem.

Often databases are not available or despite their availability some sequences are not readily found therein (Allmer et al. 2004). Therefore, *de novo* sequencing can be used to directly assign a peptide sequence to an MS/MS spectrum for which many algorithms have been developed (Allmer 2011).

1.3.2. *De Novo* Sequencing

The *de novo* sequencing problem is that an amino acid sequence needs to be assigned to an MS/MS spectrum without the use of a sequence database or other additional information. In this respect, an ion type (see Figure 1.6) needs to be assigned to the abundant peaks in an MS/MS spectrum. After that, consecutive ions of the same type differing in the mass of an amino acid needs to be established and then assigned an amino acid sequence. Hence, there should be peaks that differ in the mass of an amino acid, which indicates that they are of the same ion type. In proteomics, many algorithms are established to assign a sequence to a MS/MS spectrum (Seidler et al. 2010).

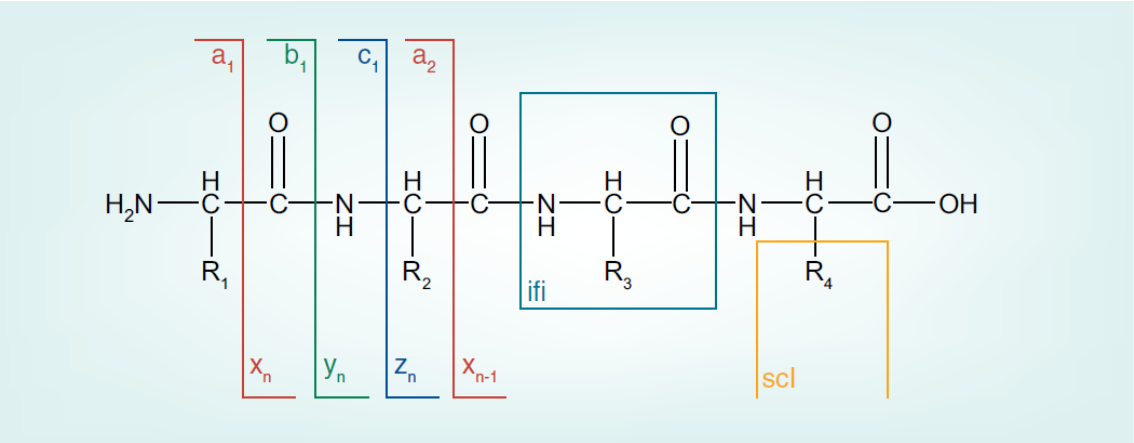

Figure 1.6. A peptide can fragment at any chemical bond in the molecule. Some frequently observed ions are named within the figure. The names of the N-terminal ions are indicated on the left of the respective fragmentation border whereas the C-terminal portions are named on the right. A subscript indicates their position within the sequence in respect to their mass. ifi: Internal fragment ion; scl: Side chain loss (Source: Allmer 2011).

One of the first approaches used to assign a sequence to a MS/MS spectrum is the brute force, or naive, approach. In brief, all amino acid sequences approximately matching the measured precursor mass are generated and scored against the spectrum. The sequence with the highest score is then accepted as the correct solution. Another approach is a spectrum graph (Figure 1.8) that is the transformation of a peak list into a graph where each m/z value is represented by a node in the graph. Nodes are connected by edges if their m/z values differ by the mass of an amino acid.

Dynamic programming is a technique that can be applied if a problem can be broken into smaller problems that, once solved, are able to solve the larger problems. Furthermore, the ability to build upon intermediate results is necessary for dynamic programming, ensuring no recalculations, which makes it especially suitable for finding optimal paths through a graph (e.g., spectrum graph), although faster heuristics exist. Genetic algorithms have been used for sequence optimization, which is in essence quite similar to the naive approach with the difference that not all possible amino acid sequences need to be generated. Instead, a small pool of amino acid sequences is generated and then optimized to best fit the MS/MS spectrum. The divide and conquer algorithm for splitting the spectrum into successively smaller subspectra until they are solvable by the naive approach and are then recursively reintegrated to solve the input spectrum. The use of hidden Markov models, as an example for machine learning algorithms, has been used to tackle the *de novo* sequencing problem. A pattern-based

algorithm has been introduced (Allmer 2011). Using these algorithms, many implementations such as PEAKS, Lutefisk, PepNovo, Unnamed, NovoHMM, SeqMS, EigenMS, AuDeNs, MSNovo, MAARIAN, PFIA, Vonode are created in *de novo* sequencing area (Allmer 2011).

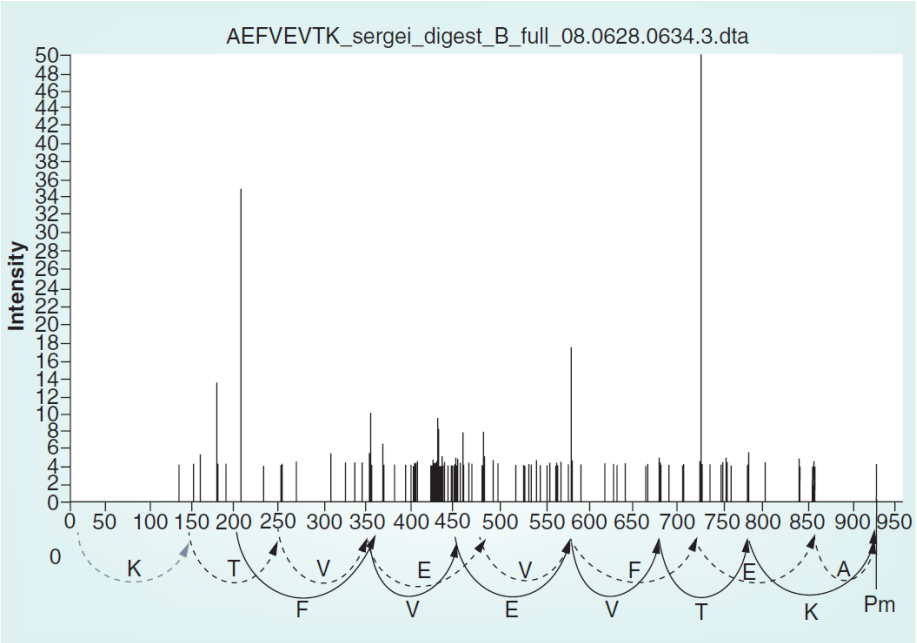

Figure 1.7. In this figure, spectrum graph approach is shown to peptide identification. MS/MS spectrum is converted into spectrum graph. Each node represents m/z of peak with score s.Edges connect nodes if mass difference of nodes corresponds to an amino acid (Source: Allmer 2011).

Currently, these *de novo* prediction algorithms store all the information in their particular, if not peculiar, formats. Hence, algorithm-specific solutions are abundantly available some listed above. However, there is no a general solution to store and share *de novo* sequencing results which diminishes the usefulness of these results. Hence, they need to be integrated and further analyzed and cannot be used directly as evidence for protein identification or sequencing at present. In order to further the field of *de novo* sequencing a standard is vitally important.

1.4. Standardization

A standard file format is a format by which specifications are made publicly available. Hence, all users can encode and read files in this format. The main goal of defining standards is to allow software to communicate with each other due to the processing data produced by any tool except having to bother about file format conversions. This is especially important to ensure transparency and interoperability and to facilitate integration and exchange from different source. In mass spectrometry-based proteomics, many standards are disclosed to eliminate these problems (www.psidev.info)(Orchard et al. 2004).

Extensible Markup Language (XML) (http://www.w3.org/XML/) is a language which provides rules to encode documents in a format which is both human and machine readable. It is created by the W3C (www.w3.org). The aim of XML is simplicity, generality, usability. Several schema systems exist to provide definition of XML based languages. In addition, schema provides validation process during which file is checked for suitability with this schema. Until now, A standards (www.psidev.info) in proteomics are created by XML technologies due to mostly represented data in the computer science, providing easy implementation, testing, and designing and human readable representation. Therefore, DNML is constructed by XML technologies.

A controlled vocabulary is a restricted list of words or terms typically used for descriptive cataloging, tagging or indexing. It is controlled because users (catalogers, taggers, indexers) may only apply terms from the list for its scoped area (its metadata value or field). It is also controlled, because only under certain specific conditions and review processes may the terms within a controlled vocabulary change or grow, and this is the responsibility of a controlled vocabulary editor or *taxonomist*, not the users. The term ‘controlled vocabulary’ is broad and covers the full range of different kinds of structures for term management. General aim of standards in proteomics inherits CV (A controlled vocabulary) representation to show element and relations between them (http://www.obofoundry.org/?sep). Hence, DNML and DNMSO controlled vocabularies are constructed to represent these. In proteomics, CV is generally represented by obo format which is ontology representation language.

Ontologies are used to capture knowledge about specific domain of interest. Ontology describes the concepts in the domain and also the relationships that hold

between those concepts. Different ontology languages provide different facilities. The most recent development in standard ontology languages is OWL from the World Wide Web Consortium (W3C). It has a richer set of operators such as intersection, union and negation. It is based on a different logical model which makes it possible for concepts to be described. Complex concepts can therefore be built up in definitions out of simpler concepts. Furthermore, the logical model allows the use of a reasoner which can check whether or not all of the statements and definitions in the ontology are mutually consistent and can also recognize which concepts fit under which definitions. The reasoner can therefore help to maintain the hierarchy correctly. In this respect, DNMSO is created by obo format which is an ontologies language created for biological and biomedical (http://www.obofoundry.org/).

Several areas of mass spectrometry-based proteomics have standardization for collections of mass spectra and MS/MS spectra in files like mzXML (Pedrioli et al. 2004), mzData (Orchard et al. 2004) and most recently in mzML (Martens et al. 2011). The Human Proteome Organization (HUPO) oversees many standard initiatives and is adopting community standards or actively develops new ones. Examples for community standards are protXML and pepXML (Keller et al. 2005) which store the result of database search algorithms. HUPO has published on the mzIdentML (Jones et al. 2012) standard to supersede pepXML and protXML (Keller et al. 2005).

As there is currently no standard to represent *de novo* sequencing results, other, not suitable formats are used to store *de novo* predictions. PEAKS, commercial *de novo* sequencing tool(Ma et al. 2003), for instance, stores results in pepXML format. The other *de novo* sequencing algorithms, are the two most popular freewares, PepNovo (Frank and Pevzner 2005) and Lutefisk (Taylor and Johnson 2001), are using custom formats which unfortunately do not provide adequate information for further downstream analyses and knowledge mining. mzIdentML (Jones et al. 2012) and pepXML (Keller et al. 2005) are created for database search results but they are not suitable for *de novo* sequencing. Hence, all these standards are bulky in order to support database search which has a lot of unnecessary elements for *de novo* sequencing. Also, they don't have many to many relations between spectra and predictions and support elements that show proof peaks of prediction. It is believed that it is necessary to develop a standard representation for *de novo* sequencing results since none of the current programs represent the minimum information needed and since standards from related fields are either flawed or not fully applicable.

XML technology is selected as the first version of this study in order to be used to represent data in the computer science, provide easy implementation, testing, and designing and human readable representation. On the other hand, XML have some limitations such as parsing, tree, namespace, context, self-description, modeling, schema, RDF, Ontology (Wilde and Glushko 2008). Therefore, Ontology base is preferred as second version of this study. Obo-format other than owlapi (http://owlapi.source forge.net/), jena (http://jena.apache.org/) is selected because of fast, compact, implemented for biological domain by obo foundry, easier implementation and flat file system, wrapped for using owlapi.

1.5. The Aim of the Study

The standard will be implemented to eliminate problems of other standards such as that there are serious design limitations of them that each result can only be linked to one spectrum source. Recently, more and more algorithms are designed that analyses multiple spectra in tandem to produce a peptide prediction (Takan and Allmer 2012). These can thus not be represented in these formats. In order to solve this problem, many-to-many relationship between spectra and predictions will be created. Another problem is that these spectra files are huge (sometimes several gigabyte) it does not help the exchange in-between laboratories but only moves the problem from a standardization problem to a technical problem. Since even 80% of spectra may remain unassigned, it leaves two options, one to link to an mzML or mzXML file and the other to include the spectrum in the DNML or DNMSO file. This leads to the possibility of exchanging only one file containing all relevant information. These formats have been designed with the representation of the best result in mind. DNML or DNMSO will have no such limitation and all results produced by a *de novo* prediction algorithm can be represented. In addition to offering modification possibilities as others does, it will be provided that all current PTMs as a separate XML file. Furthermore, DNML will offer the ability to represent gaps which is not possible in other formats and thus excludes some *de novo* sequence results for example from Lutefisk. It is argued that analysis is not part of the *de novo* result or any software result representation and thus will not be part of this representation. It is urged that visualization should not be part of any data modeling attempt and that it needs to be handled separately. Redundant or easily calculated information should also not be part of a serious attempt to

standardization since it makes the format more rigid and unnecessarily inflates the file size.

A programming interface will have been developed since the missing of proper APIs is identified as another obstacle, introducing a needlessly high learning curve for developers about adding, deleting, transferring data, and converting between selected standards, merging. What is more, a lot of attention is unfortunately diverted to unnecessary file handling which could be used to improve algorithms. The input file handling problem will be removed by providing an API which allows developers to read spectra from the two most important standards, mzXML and mzML. XML technology will be selected as the first version of this project in order to that mostly used to represent data in the computer science, provide easy implementation, testing, and designing and human readable representation. Ontology base will be preferred as second version of this study due to better representation of domain and easier support for reasoning and other soft computing and artificial intelligent facilities.

Graphical user interface will have been created for visualization and analysis using this advanced programming interface. In this respect, predictions, spectra and relations between them will be shown easily. Spectra can be represented by charts .Also, all operations such as merging, adding, deleting, transferring data and converting can be performed with GUI.

In brief, the outcomes of this study are

1. To provide many-to-many relationships between spectra and predictions, exchange and merging, showing all results, PTMs, compact, no using proprietary formats with a new standard.
2. To develop an advanced programming interface. It is standard, compact, Modular, easily extensible and also have read, write, create, convert, supports current standards facilities.
3. To create Graphical user interface for visualization, analysis, merging, adding, deleting, transferring data, converting, seeing spectra, predictions and relations between them.

CHAPTER 2

MATERIAL AND METHODS

2.1. Ontology

Semantic web technologies enable structuring of documents and thus allow unstructured documents to be transformed into a “web of data”. The semantic web is based on a simple idea. However, it has many problems such as realization due to vastness, vagueness, uncertainty, inconsistency, and deceit (Shadbolt et al. 2006; Smith et al. 2005).

The World Wide Web Consortium (W3C) created many standards such as Resource Description Framework (RDF), RDF Schema (RDFS), and Web Ontology Language (OWL) to support the semantic web in order to solve these problems. An ontology is a definition of a representation vocabulary (formally) for shared domain of discourse, definitions of class, relations, functions, and other objects (Gruber 1993). Obo format programing interface, API, (https://code.google.com/p/oboformat/) is used in this study for reading and writing DNML output which is represented by this ontology.

In this study, Protégé (protege.stanford.edu) and OBO-Edit (http://oboedit.org/, version is 2.1) were used as ontology editors in order to create ontology.

2.2. XML

Particular attention is paid to using standard JAVA™ libraries for XML parsing for compatibility reasons. The document object model (DOM) was found unsuitable since many result files are large and since DOM requires the entire file to be in memory after parsing. The simple API for XML (SAX) allows sequential access to XML elements and has a lower memory footprint and was therefore selected for this project.

Specifically, the JAXB technology has been selected for XML parsing in the DNML API. These tools marshal Java objects into XML or unmarshal XML into java objects. Hence, XML mapping with the XJB tool allowed for easy conversions between

XML and Java classes and sped up the design process by reducing the duplication of code.

JmzReader GUI version 1.2.7 (http://code.google.com/p/jmzreader/) was used for getting mzXML and mzML data. It uses JAXB technology (stream base) so that it can retrieve data from huge files (more than 1 GB).

Oxygen (http://www.oxygenXML.com/) was used for XML editing because of the availability of a trial version.

Two different libraries are used for parsing mzXML files. The first is the Jrap library which caused many problems for us but which enables us to retrieve spectra modeled as links in DNML format from the mzXML format. The other is the Jmzreader library which uses the JAXB JAVA™ technology which integrated seamlessly and provides the link between DNML and mzML.

2.3. Programing Language

In this study, for programming, the Java™ language was selected because it seamlessly allows interoperability to have many libraries and to have a huge support community.

The Eclipse IDE was selected on the basis of available plug-ins and the extensive user support. Download link is http://www.eclipse.org/ (Eclipse IDE Version 3.7)

2.4. Other Technologies

Maven is a build automation tool which is used for java projects similar to the ant tool. Maven has been started by the jakarta project. Maven can download java libraries dynamically from maven repositories. Maven further provides artifacts which can help initiate different project types. an eclipse plugin is available which we used in this study http://www.eclipse.org/m2e/.

Regular expression, part of the standard JAVA™ library, were mostly needed for con-version facilities for example to parse LutefiskXP and PepNovo *de novo* predictions. RegexBuddy was used for development and testing of regular expressions (http://www.regexbuddy.com/,version 5.2)

Version control is a management system which handle multiple versions of documents such as source code to allow for collaboration. GIT, which is provided on Bitbucket (https://bitbucket.org/), was used during development as a version control system.

eUML (http://www.soyatec.com/euml2/) was utilized to support The Unified Modeling Language (UML) visualization during the development process. Visual Paradigm (http://www.visual-paradigm.com/, version 8.2), was used to create some UML diagrams and to enable round-trip engineering. Another tool was argoUML (http://argouml.tigris.org/ , version 0.34).

All figures in thesis have been created by Microsoft Paint and PowerPoint.

2.5. Test Data

For testing, several datasets were used. First dataset was measured with the Applied Biosystems MDS SCIEX 4000 ESI Q-Trap and the Thermo Scientific LTQ XL Linear Ion Trap ESI mass spectrometers (MS) with CID fragmentation, respectively. Prepared peptide mixture was introduced to the mass spectrometers directly without any prior chromatography steps. Herein, parameters including filling time, activation time, collision energy (CID), TIC and cycle numbers were alternated to obtain spectra with different spectral. MzML dataset was obtained from http://www.psidev.info/mzml/ . LutefiskXP (open source, http://www.hairyfatguy.com/lutefisk/ (version is 1.07)), PepNovo (open source, http://proteomics.ucsd.edu/Software/PepNovo.html, build time: 2012.04.23), Peaks (commercial, output is PepXML, http://www.bioinfor.com/) are popular *de novo* sequencing algorithms. In this study, these *de novo* sequencing algorithms were used for creating other data sets which were de novo results. What is more, API was used in COMAS, another *de novo* sequencing algorithm created by JLAB (http://bioinformatics.iyte.edu.tr/) in Izmir institute of technology, in the visualization tool which was developed in this project for testing. All these *de novo* sequencing algorithms run with the same parameters where fragment tolerance as 0.3000, pm tolerance as 0.8000 were used. In this respects, these datasets converted to DNMSO files by API. Hence, functionality, efficiency, usability of our DNMSO API was tested by writing, merging, reading, converting functions.

CHAPTER 3

DNML: *DE NOVO* MARKUP LANGUAGE

3.1. The Standard and Model

The *de novo* markup language (DNML) has been developed to create a standard for the representation of *de novo* sequencing results. First and foremost, it is ensured that all necessary, but as little unnecessary data as possible, is represented in the format. Another issue that it is identified to be crucial to the acceptance of a standard initiative is the availability of advanced programming interfaces (API) to read and write the standard. Many standards fail to offer an API and if they do only for reading of the standard. To increase the acceptance of DNML it is gone well beyond this and offer reading, writing, and conversion facilities. But most importantly, the API allows for the creation of files and can thus be used directly in any software that performs *de novo* sequencing. This removes the need to develop reading and writing functionality for all developers of *de novo* sequencing algorithms, a mundane task which is often neglected and therefore often produces peculiar outcomes. Finally, it is also included reading of mzXML and mzML information in the API so that any developer of *de novo* sequencing software can completely focus on the algorithm. In the following the schema, the API is introduced.

3.1.1. XML schema

The W3C developed DTDs (A Document Type Definition) and XML schema languages to be used to allow the validation of XML formatted documents. Since today DTDs seem to be deprecated, a pure XML schema is developed without any reference to DTDs. During the creation of the XML schema, object oriented programming techniques were used for solving repeated, non-extendable data problems.

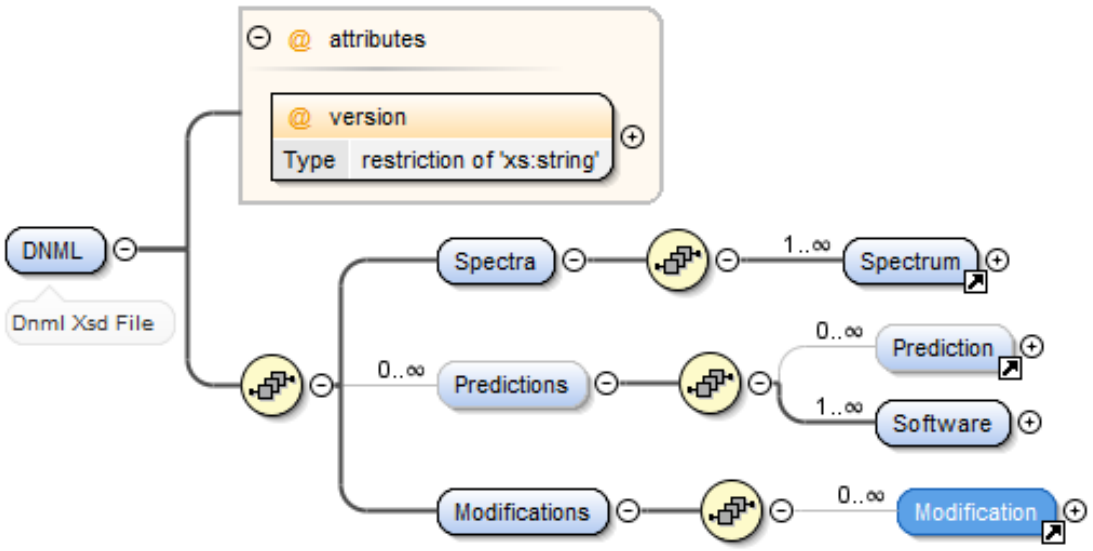

Figure 3.1. Depicts the DNML root element and the contained Spectra, Predictions, and Modifications elements. The XML tree can be further expanded to reveal that Predictions, for example, contain Prediction elements and a Software element.

The root of the DNML schema and its directly contained elements are shown in Figure 3.1. Spectra represent the underlying data source for *de novo* predictions and is modeled in two different ways. One way minimalizes file size, and the other directly links to spectra contained in files of the appropriate standard.

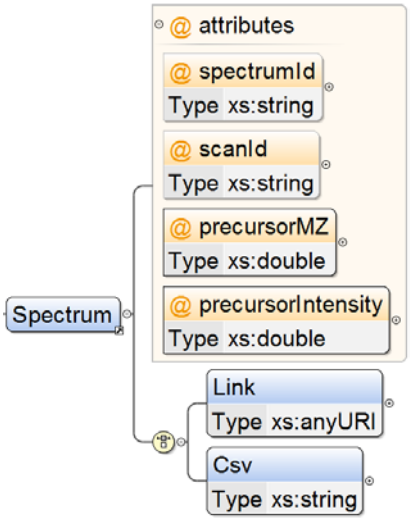

Figure 3.2. Shows how the raw data source, the spectrum, is modeled. Spectrum has attributes such as id, precursor m/z, precursor intensity. The peaks in the MS/MS spectrum can either be provided as a link to a file containing spectra in mzML or mzXML standard, or as contained data.

Because the files in mzXML and mzML format quickly become huge, it is difficult to share them. Therefore, spectra can be contained within the DNML standard optionally. Different data encodings are provided to additionally reduce the size of the DNML file.

DNML offers the opportunity to store spectrum data in several formats such as Base64, CSV and as links to accepted formats. API further allows the future incorporation of additional data formats such as compressed formats due to the extensible design (Figure 3.2).

When a spectrum is provided as a link to mzXML or mzML formatted files, the DNML file remains lean and the API allows the linked data to retrieve.

While developing DNML the problem of merging multiple files has been taken into consideration. In the library, commands are available which enable DNML to merge and split its formatted files.

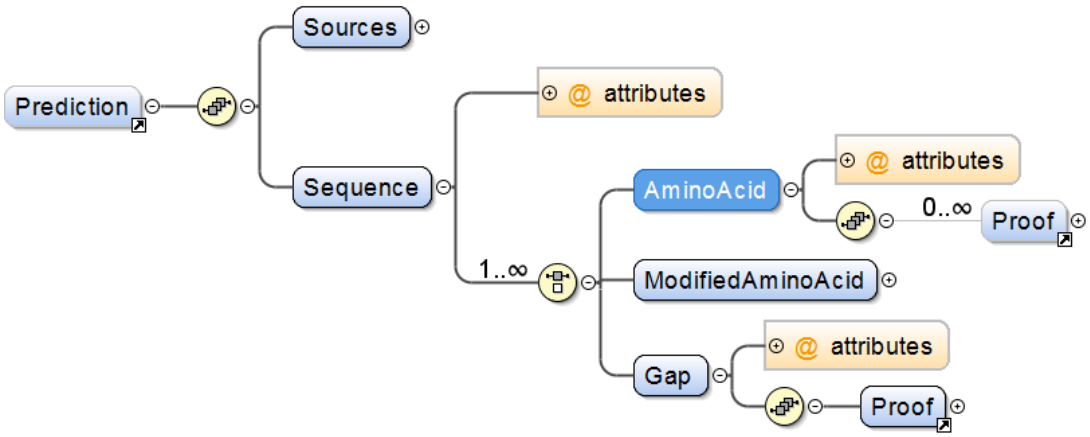

Figure 3.3. The Prediction element contains Sources, which defines the spectra that led to the prediction and Sequence, which contains sequence and confidence attributes.

DNML format is the first one which allows a many-to-many mapping between predictions and spectra. This has become important since many *de novo* sequencing algorithms now use multiple spectra from different sources to produce a better prediction (Figure 3.3). Since a prediction can contain stretches where no amino acids can be assigned, it is necessary to model gaps. DNML standard is the first which enables this, but this a step is taken further by enabling object oriented creation of sequences which are composed of three different types of elements, amino acids, modified amino acids and gaps.

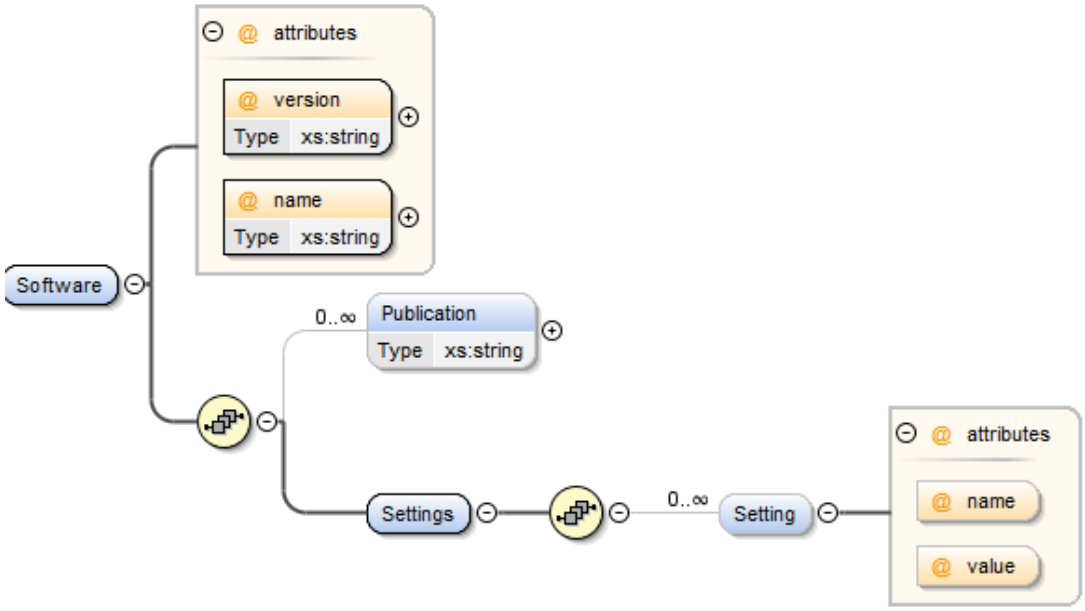

Figure 3.4. Software represents the tool that made one or multiple *de novo* predictions. Publication element is used to add publications of that software.

An additional novelty is that each sequence element can be amended with peaks proving the existence and with a positional confidence.

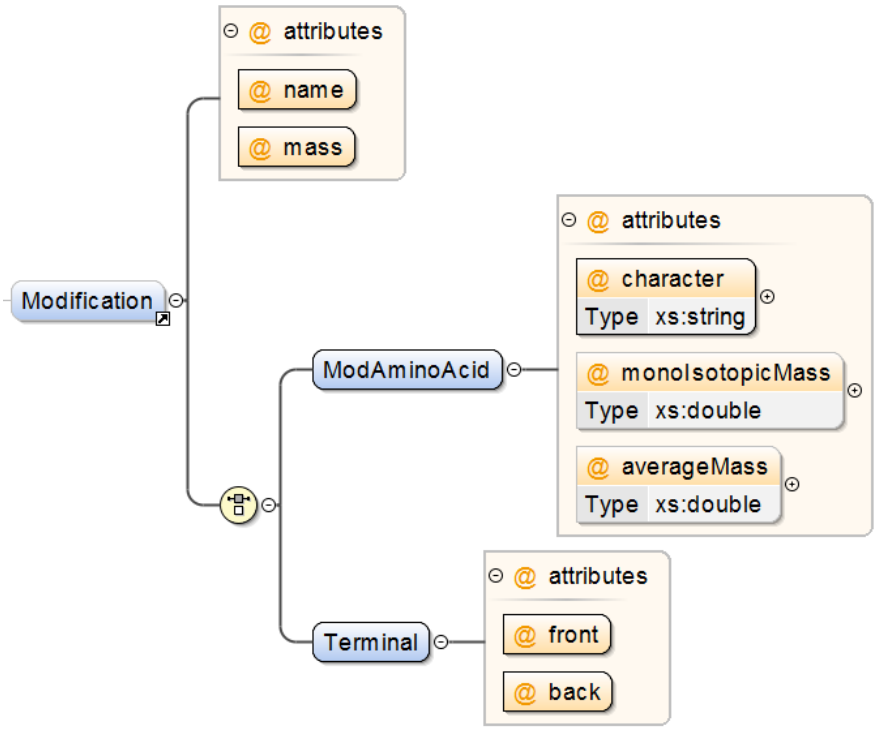

Figure 3.5. A post translational modification (PTM) is modeled as a change to an amino acid and a file with all currently known PTMs is provided to simplify handling of PTMs. Terminal modifications for peptides are also possible.

The user's software settings need to be provided since different settings can lead to significantly different predictions. The model allows different “Predictions” to have different settings so that the information cannot be repeated for each Prediction (Figure 3.4).

The proper modeling of post translational modifications PTMs is important today but its importance will increase in the near future when more algorithms will be able to predict unexpected PTMs. It is allowed user specified PTMs but also provide a list of about 100 known modifications. Another important innovation of DNML is that predictions can be complemented with the underlying proof for the sequence is much better reflected in the format (Figure 3.3). The proof which is not completely expanded in Figure 3.3 links to the peaks in the MS/MS spectrum which defines a particular sequence element. A sequence element can be an amino acid (normal), a modified amino acid, or a gap which means an unknown range in the MS/MS spectrum which is not assigned an amino acid (Figure 3.5).

3.2. The Implementation

The DNML library provides a programming interface. Adding, deleting, merging, transferring data, and converting between selected standards are supported by the API. In addition to this, service, command classes can be added at runtime by using the JAVA™ resource bundle which creates a great flexibility and makes the library easily extendible and modular (Figure 6). The structure of the API has several layers:Façade, Controller, Services, and Model (Figure 6).

The façade factory, a static singleton, is the main interface to the façade layer and pro-vides the only means to communicate with the layer. Inputs are taken as a string array. In this way, the program can run in any environment. When data is entered, the façade layer forwards inputs to the control factory which creates a command. The façade’s output is a DNML domain object, created using the command layer.

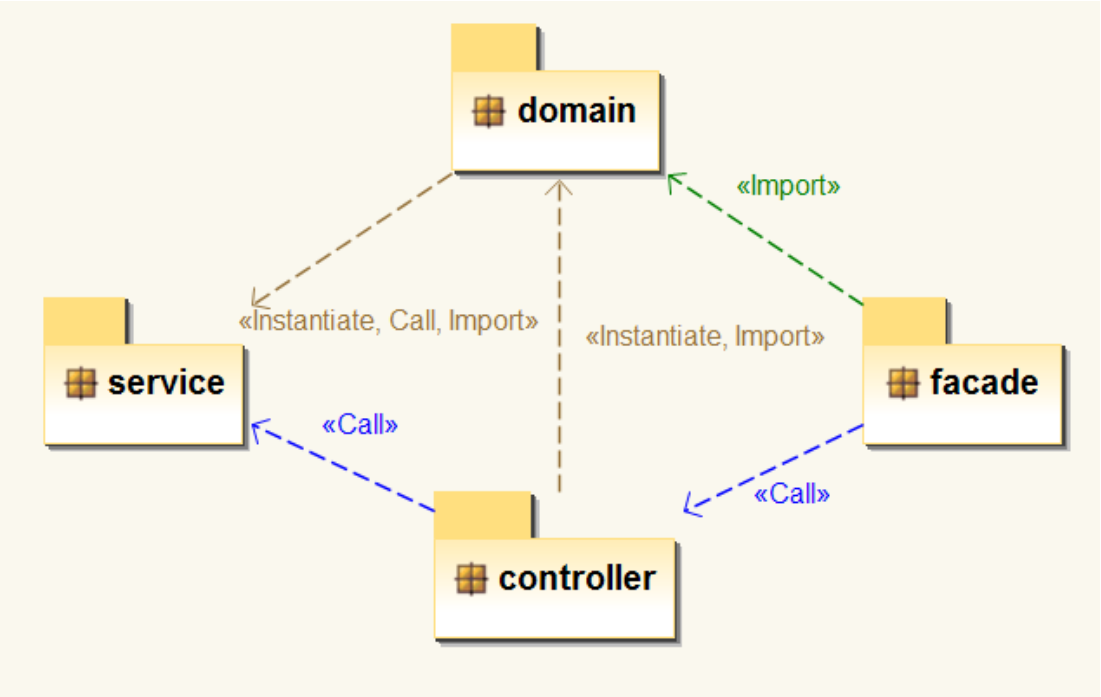

Figure 3.6. The DNML library has four packages. The façade package handles in-put and sends it to the controller. The controller finds, creates, and executes commands. The service package handles do-main tasks.

Communication with the controller layer is established through its static factory singelton. The factory creates the appropriate command based on the input and executes it. Each command must implement the command interface in order to be compatible. This interface has two functions; execute and setArgs. The execute function executes the command and returns a DNML domain object. The setArgs method allows the customization of a command. The factory includes a property file to find command location by command name at runtime. This structure is very important for extensibility and modularity. Hence, when a new command needs to be added, there is no need to rebuild the library. The control layer submits the task to the service layer after the process is started by command.

In order to ensure modularity of the service layer, all services are created independently. As with the previously mentioned layers, the service layer factory is a static singleton. The factory creates suitable service depending on the given input and in collaboration with the service layer ensures their execution. Each service must implement the services interface which has an object container for handling input as a string array. The first operation is parsing the input using regular expressions to check whether the input is suitable in respect to security and conformity to the service

factory's expectations. When successful, the requested service is run employing the given input. The output is again modeled as an object container which is important for the concept of modularity. Thus, it provides great flexibility for future extensions. This flexibility is also implemented on the mapping between XML and parsing classes. The domain layer (not shown) contains classes which, depending on XJB for JAXB, seamlessly propagate changes between the DNML schema and the parser. This technology creates the flexibility allowing us to quickly respond to the new developments in the field which may entail changes to the DNML schema.

3.3. Comparison

Often corruption is realized while parsing files or found files that did not exactly adhere to their target standards. This is clearly undesirable and most likely due to the redundant development of APIs by different vendors. The API provided removes this problem since all functionality, reading, creating, and writing are available thus removing the possibility of creating files that do not validate or are corrupted. This also enhances the confidence of developers of downstream tools, building on data provided in DNML format. The link provided to spectra in mzXML and mzML format can be accessed by id or can be extracted as text further enhances the ease with which API can be used. API thus allows access to all necessary input for *de novo* sequencing and all output thus completely removing file handling from the developers of *de novo* algorithms. Since many spectra have been processed by existing *de novo* prediction tools and since those results are not available in DNML format provided conversion facilities to the currently most prominent free tools, LutefiskXP and PepNovo but further conversion facilities for PEAKS.

The DNML schema and the fully fledged API are the main outcomes of this study. Especial attention is paid especial attention to extensibility of the API and most parts of the code allow for runtime binding of additional functionality thus making it possible for developers to easily extend our API. The API, which includes conversion functionality, can be downloaded from http://bioinformatics.iyte.edu.tr/dnml. The conversion functionality can also be directly accessed at http://bioinformatics.iyte.edu.tr/dnml/convert2DNML.php.

CHAPTER 4

DNMSO: *DE NOVO* MS ONTOLOGY

4.1. The Standard and Model

Mass spectrometry-based proteomics depends on computational tools for data analysis. There are three possibilities to assign sequence to mass spectra. One of them, *de novo* sequencing, directly infers the sequence from MS/MS spectra. Although many tools have been developed for *de novo* sequencing, they are not very effective. The representations of results vary from algorithm to algorithm and integration of various tools to do Meta predictions is thus tedious also doesn't exit. Each new tool also has to develop spectra import and result file representations.

In order to enable the easy implementation of Meta predictors and in order to simplify file handling, a standard representation for *de novo* prediction result is offered: *De Novo* MS Ontology (DNMSO). The standard with an application programming interface in Java, which handles reading of various spectra file formats, reading of DNMSO, writing, creating and editing of DNMSO, and several conversions from existing *de novo* results to DNMSO, is amended.

DNMSO and DNML differ in that it uses ontology which eliminates XML problems such as parsing, tree, namespace, schema (Wilde and Glushko 2008) and provides artificial intelligence facilities such as reasoner to analyze data in future. What is more, it is released that DNML has some implementation bugs and architecture problems. Therefore, it is redesigned and bugs are fixed.

4.1.1. Ontology

Ontologies are used to capture knowledge about some domain of interest. Ontology describes the concepts in the domain and also the relationships that hold among those concepts (Alterovitz et al. 2010). Different ontology languages provide different facilities. The most recent development in standard ontology languages is OWL from the World Wide Web Consortium (W3C). It has a richer set of operators - e.g. intersection, union and negation. It is based on a different logical model which makes it possible for concepts to be dened as well as described. Complex concepts can therefore be built up out of simpler concepts(McGuinness and Harmelen 2004).

Furthermore, the logical model allows the use of a reasoner (Wang and Zhang 2004) which can check whether all of the statements and definitions in the ontology are mutually consistent and can also recognize which concepts fit under which definitions. The reasoner can therefore help to maintain the hierarchy correctly. This is particularly useful when dealing with cases which classes can have more than one parent.

Due to all these feature and also XML limitations such as parsing, tree, namespace, schema (Wilde and Glushko 2008), DNMSO is implemented by ontology other than DNML which is XML base. During the creation of DNMSO ontology, object orient programming techniques were used to solve repeated, non-extendable data problems. DNML ontology has many parts which explain below.

```
[Term]
id: DNMSO:000005
name: software
property_value: hasSettings DNMSO:000006
property_value: hasSoftwareName "LutefiskXP" xsd:string
property_value: hasSoftwareVersion "v1.0.4" xsd:string
```

Figure 4.1. in this figure, sample ontology element is shown in obo format. Each element has unque id as standard.

DNML has unique id for each element which is represented by XML schema. However, universal uniqueness is not an easy task. Hence, there are no standard tools to provide it in XML. Therefore, this mechanism is needed to be implemented manually in XML base standards such as DNML. On the other hand, ontology has this mechanism as standard property. DNMSO is no needed to implement it in data model. Id representation in ontology elements is shown in figure 4.1.

In DNMSO ontology, the root element is the DNMSO element. It represents mass spectrum that was processed. It also contains all possible *de novo* predictions, modification for this spectrum. Spectrum itself may or may not be grouped with further Spectrum for example a mass spectrometric analysis of a number of peptides in a given sample.

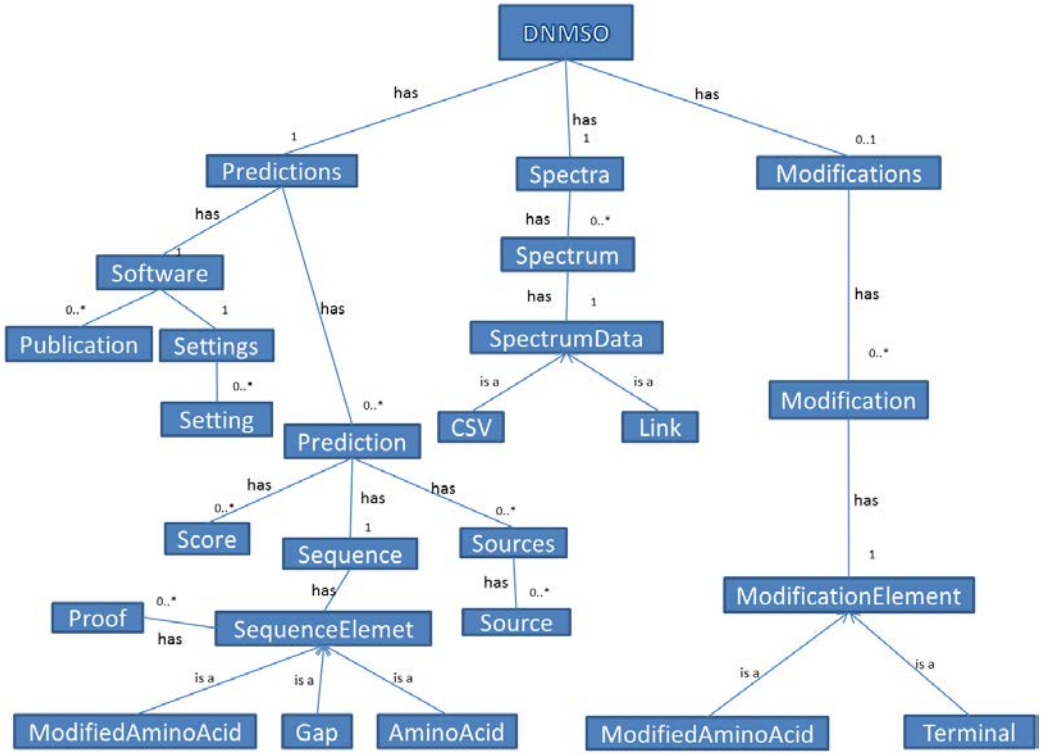

Figure 4.2. Depicts the DNMSO root element and the contained Spectra, Predictions, and Modifications elements. The Ontology can be further expanded to reveal that Predictions, for example, contain Prediction elements and a Software element.

The Spectrum element is described by its properties. Spectrum is represented by id in DNML. However, it creates uniqueness problems due to different dnml file having the same spectrum id. To eliminate, Spectrum is defined by file name and scan id and connected using the same parameters by source element in prediction element. The Spectrum represent the underlying data source for *de novo* predictions and are modeled in two different ways, one to minimalize file size, and one directly linking to spectra contained in files of the appropriate standard. Because files in mzXML and mzML format quickly become huge, it is difficult to share them and therefore Spectrum can be contained within the DNMSO standard optionally. Different data encodings are provided to additionally reduce the size of the DNMSO file. DNMSO offers the

opportunity to store spectrum data in several formats, such as CSV and as links to accepted formats. API further allows the future incorporation of additional data formats such as compressed formats due to extensible design (Figure 4.3). When a spectrum is provided as a link to mzXML or mzML formatted files, the DNML file remains lean and the API allows the linked data to retrieve.

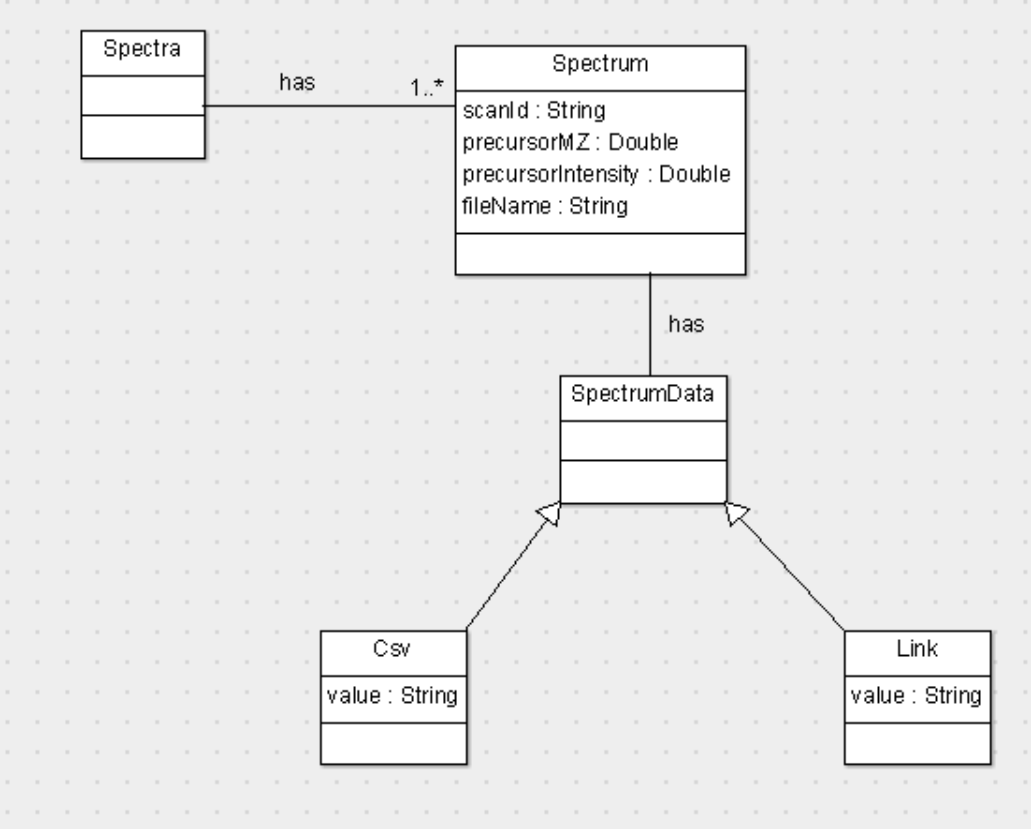

Figure 4.3. shows how the spectrum is modeled. Spectrum has attributes such as scan id, filename, precursor m/z, precursor intensity. The peaks in the MS/MS spectrum can either be provided as a link to a file containing spectra in mzML or mzXML standard, or as contained data.

Predictions are able to be used as group of predictions in the same software. Therefore, software element is not part of prediction element as DNML, which solves space problem because all predictions with the same software need to be written with it, which creates redundancy. Prediction has assumedCharge which are important for the result of *de novo* sequencing algorithms other than DNML. To eliminate, it is provided that software element is part of Predictions element in DNMSO (Figure 4.4).

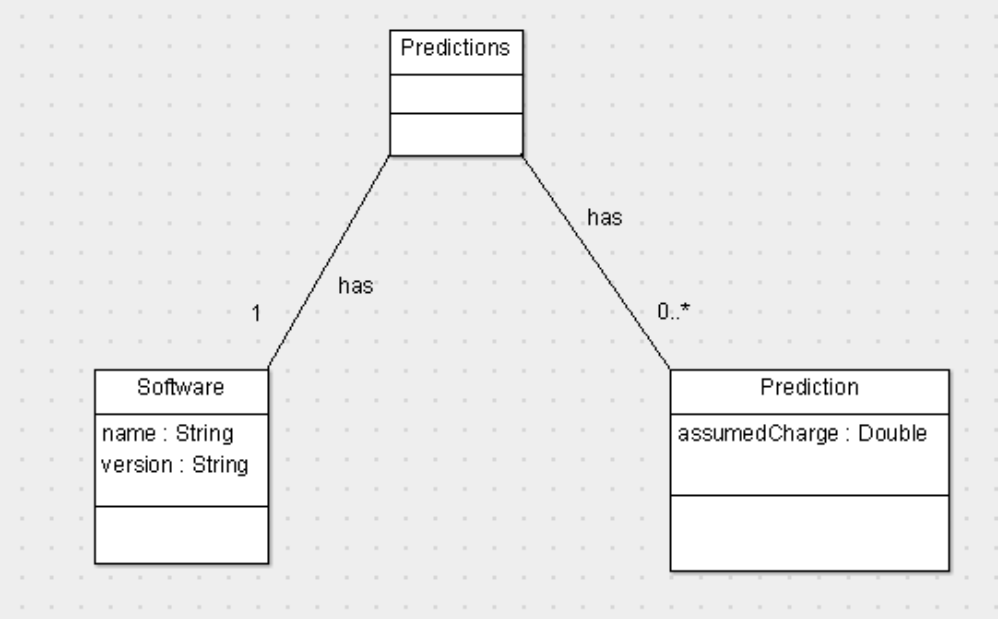

Figure 4.4. The Predictions element contains two elements. The Software element is useful to group predictions from different software tools. It is of note in this context that Predictions may thus be contained in Source multiple times.

The Software element names the tool that is used to make the prediction. Optionally, further settings can be specified in the Settings element contained in the Software element. The name attribute of the Software element simply gives the name of the software used. The settings element, contained in the Software element, may appear multiple times while specifying different parameters that are set during the processing, specific to the software used. Most tools need the specification of the mass error of the instrument for example. The user's software settings need to be provided since different settings can lead to significantly different predictions. The model presented in this study allows different Predictions to have different settings thus the information is not repeated for each Prediction (Figure 4.5).

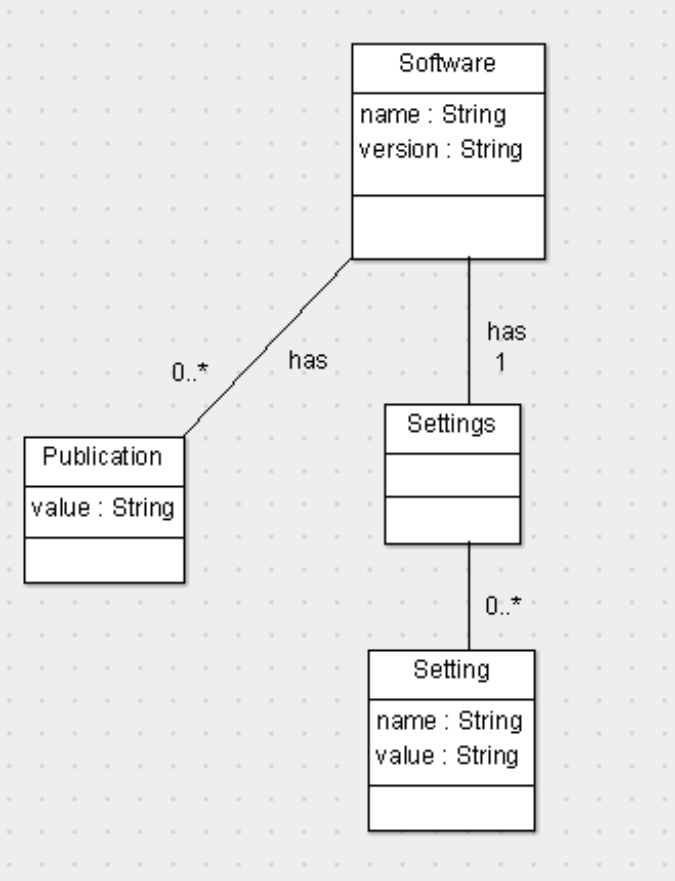

Figure 4.5. Software represents the tool that made one or multiple *de novo* predictions. Publication element is used to add publications of that software since trouble sometimes are experienced when searching for the first publication of an algorithm.

DNMSO format allows a many-to-many mapping between predictions and spectra. This has become important since many *de novo* sequencing algorithms now use multiple spectra from different sources to produce a better prediction (Figure 4.3). Sources element also is grouped by boolean parameters as merged or not merged in DNMSO. If it is selected as merge, all intensities in spectra is added using some threshold; this property is important for represent some algorithms results such as PepNovo plus which use added intensities for calculation. Since a prediction can contain stretches where no amino acids can be assigned, it is necessary to model gaps. DNMSO standard enables this, but this a step is taken further by enabling object oriented creation of sequences which are composed of three different types of elements, amino acids, modified amino acids and gaps. DNMSO has the score parameter in prediction except DNML. In this reason, extra score parameters (for example, Lutefiks parameters are Pr(c), PevzScr, Quality, IntScr, X-corr; PepNovo parameters are RnkScr, PnvScr) can be added other than DNML. What is more, Each sequence element can be

amended with peaks proving the existence and with a positional confidence. See Figure 4.6 for an overview of where these elements fit into the DNMSO ontology.

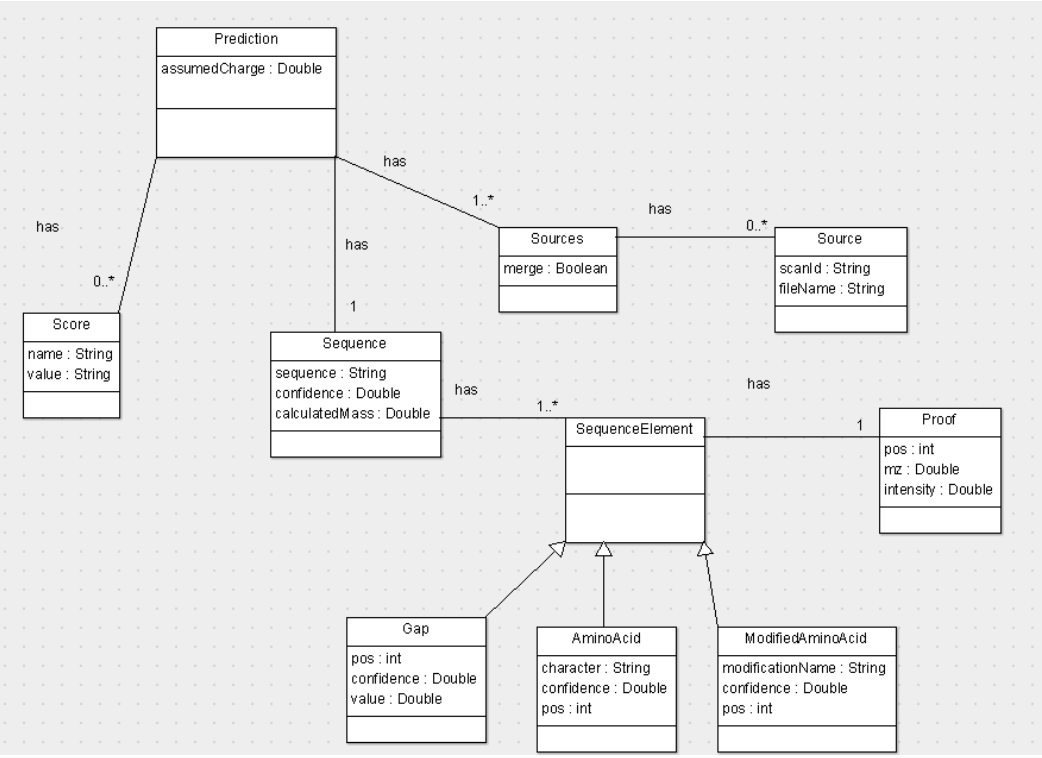

Figure 4.6. The Prediction element contains sources which define which spectra led to the prediction, Sequence which contains sequence and confidence attributes, Score which has name and value parameters and different score parameter in prediction algorithms are represented.

The most interesting information about any *de novo* experiment is the full length amino acid sequence prediction for a mass spectrum. This prediction is encoded in the Sequence. The Sequence directly accepts the predicted full length sequence as an attribute. This information is redundant since within the Sequence the full sequence will be provided with additional information. However, in many cases the sequence is the only information needed and it may be convenient to be able to access it directly. Sequence element has calculatedMass element which is important for the result of predictions algorithms other than DNML.

The Sequence contains the sequence split up into individual building blocks called Sequence element. A sequence element can be an amino acid (normal), a modified amino acid, or a gap which means an unknown range in the MS/MS spectrum which is not assigned an amino acid. The position within the Sequence elements has to be specified in the position attribute the index is not zero based. So the first amino acid

is at position one. The confidence attribute defines the level of confidence that is given to the amino acid at this position in the sequence. The confidence attribute states how sure this fining is.

The Proof element may contain the information for the peak that defines the amino acid, but it may also take supporting peaks such as the presence of higher charged supporting ions, present as peaks, sister peaks such as a-x, b-y, and c-z or brother peaks such as H2O or NH3 losses. The Proof element may be specified multiple times within the Sequence element. Each proof for an amino acid may be listed. A confidence for a given proof can be specified in the confidence attribute. The mz attribute takes the m/z value for the supporting peak, if it cannot be associated by the peak attribute. The peak attribute may contain a link to a specific peak in the Spectrum element.

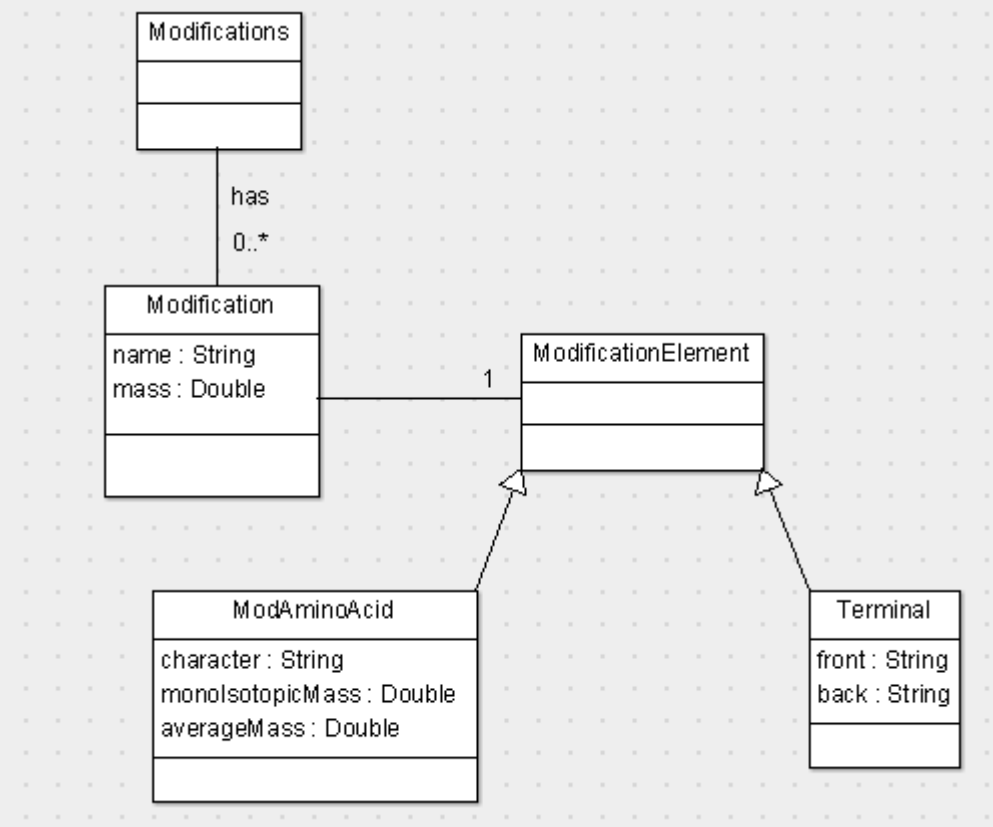

Figure 4.7. The proper modeling of post translational modifications PTMs is important today but will increase in importance in the near future when more algorithms will become able to predict unexpected PTMs. It is allowed user specified PTMs but also provide a list of about 100 known modifications. A post translational modification (PTM) is modeled as a change to an amino acid and a file with all currently known PTMs is provided to simplify handling of PTMs. Terminal modifications for peptides are also possible.

Amino acid may be predicted to be modified, which can be reported in the modification element. The property name simply gives the name of the proposed modification (i.e. glycolysation). The mass attribute gives the mass difference that has been found to modify the amino acid. The modification element may only be used once within Modifications element. If there are several possible modifications, multiple Prediction elements have to be created, each specifying a different prediction (Figure 4.7).

4.2. Implementation

The DNMSO library provides a programming interface. Adding, deleting, merging, transferring data, and converting between selected standards are supported by the API. In addition to this, service, command classes can be added at runtime by using the JAVA™ resource bundle which creates great flexibility and makes the library easily extendible and modular. The structure of the API has several layers, Façade, Controller, Services, and Model.

Each layer is controlled by factories in DNMSO. Each DNML factories are implemented by using singleton pattern which is static class. Static classes are always loaded in memory when program is running. Therefore, it cause memory problem. As a solution, it is provided that all factories in DNMSO are changed static classes into normal classes.

The façade factory is the main interface to the façade layer and provides the only means to communicate with the layer. Inputs are taken as a string array. In this way, the program can run in any environment. When data is entered, the façade layer forwards inputs to the control factory which creates a command. The façade's output is a DNMSO domain object, created using the command layer.

Communication with the controller layer is established through its factory. The factory creates the appropriate command based on the input and executes it. Each command must implement the command interface in order to be compatible. This interface has two functions; execute and setArgs. The execute function executes the command and returns a DNML domain object. The setArgs method allows the customization of a command. The factory includes a property file to find command location by command name at runtime. This structure is very important for extensibility

and modularity. Hence, when a new command needs to be added, there is no need to rebuild the library. The control layer submits the task to the service layer after the process is started by command.

As with the previously mentioned layers, the main element of the service layer is a factory. Service factory is divided into two in DNMSO : SpectraFactory and PredictionFactory. DNMSO is concentrated on prediction algorithm. Therefore, spectra shouldn't be obtained without prediction. This DNML function in is deleted in DNMSO. In order to ensure modularity of the service layer, all services are created independently. The factory creates suitable service depending on the given input and in collaboration with the service layer ensures their execution. Each service must implement the services interface which has an object container for handling input as a string array. The first operation is parsing the input using regular expressions and to check whether the input is suitable in respect to security and conformity to the service factory's expectations. When successful, the requested service is run employing the given input. The output is again modeled as an object container which is important for the concept of modularity. Thus, it helps us to provide great flexibility for future extensions. PepXML reading facility is added in DNMSO which differ from DNML. Some *de novo* sequencing algorithms such as PEAKS use this output format. Therefore, these algorithm outputs are able to be converted to DNMSO. PepXML reader package is embedded in DNMSO API. DNMSO support the first version of this study, DNML. DNML reader is embedded into DNMSO. It is good for someone if upgrade operation from DNML into DNMSO is wanted. DNMSO validation operation is provided by obo format facilities. This operation is good for data integrity and consistency. The same operation is used in DNML by XML schema.

The domain layer contains classes which seamlessly propagate changes between the DNMSO ontology and the parser. This technology creates the flexibility allowing us to quickly respond to new developments in the field which may entail changes to the DNMSO ontology. Obo format is used to represent DNMSO ontology. It is created due to running with biological data. What is more, it is flat file format other than XML, which provides elimination of problems. There is a simple and efficient library (https://code.google.com/p/oboformat/) to manipulate it, which is important for faster process. In addition, it is easily converted to owl by tools (http://code.google.com/p/owltools/). DNMSO domain is different from DNML. DNMSO is a graph representation which provides more flexibility and easier

implementation. However, some implementation techniques are different from DNML due to ontologies. Hence, JAXB XML technologies which have many good advantages such as map between class and XML are not usable in DNMSO.

4.3. Compression

Often corruption is realized while parsing files or finding files that do not exactly adhere to their target standards. This is clearly undesirable and most likely due to the redundant development of APIs by different vendors. The API removes this problem since all functionality, reading, creating, and writing are available thus removing the possibility of creating files that do not valid or are corrupted. The link provided to spectra in mzXML and mzML format which can be accessed by id or can be extracted as text further enhances the ease with which the API can be used. The API thus allows access to all necessary input for *de novo* sequencing and all output thus completely removing file handling from the developers of *de novo* algorithms. Since many spectra have been processed by existing *de novo* prediction tools and since those results are not available in DNMSO format conversion facilities are provided to the currently most prominent tools, LutefiskXP, PepNovo, PEAKS.

Especial attention is paid to extensibility of the API and most parts of the code allow for runtime binding of additional functionality thus making it possible for developers to easily extend The API.

DNMSO main difference form DNML is that it uses ontology which eliminates XML problems such as parsing, tree, namespace, schema (Wilde and Glushko 2008) and provides artificial intelligence facilities such as reasoner to analyze data in future. What is more, it is released that DNML has some bugs and architecture problems. To eliminate this problem, it is redesigned and bugs are fixed.

CHAPTER 5

VISUALIZATION AND ANALYSIS

DNMSO standard format is released for peptide identification, which can represent the output of *de novo* sequencing algorithms. The standard with an application programming interface in Java which handles reading of various spectra file formats, reading of DNMSO, writing, creating and editing of DNMSO, and several conversions from existing *de novo* results to DNMSO, is amended.

In order to allow the experimental proteomics community to analyze data stored in the DNMSO standard, Graphical User Interface is developed a, DNMSO Analyzer, an open source graphical user interface, to provide some facilities such as reading of various spectra file formats, reading, viewing, summarizing of DNMSO, and several conversions from existing *de novo* results to DNMSO in DNMSO Analyzer.

The laboratory scientists are the main users of the DNMSO Analyzer which does not require proteomics support to get started. The Analyzer can be simply downloaded and run with no complex setup procedures. Crucially, DNMSO Analyzer aims to provide intuitive and useful views of DNMSO output.

The DNMSO Analyzer provides several different methods such as viewing, summarizing, reading, several conversions from existing *de novo* results to DNMSO. Main functionality of the program is explained below.

When the program opened, it is seen that there are menu upper part. In the menu, file submenu provides open, save and close DNMSO file functionality. What is more, setting submenu in the file submenu is used to set program properties such as search tolerance, significant figure. Exit submenu menu close the program. Edit submenu is used to delete predictions, prediction which are shown in tree view section. In addition, tree view section is expanded or collapsed by expand all, collapse all commands. In operations sections, different file formats can be converted to DNMSO by convert command and DNMSO files are merged by merge command. Help submenu provides some information about how to use this program.

On the left panel in the program, there is tree view representation with summarized information for opening file. In there, search text box is created to find

suitable element in opening files within given threshold value which is configured in setting menu. A general representation is shown in Figure 5.1.

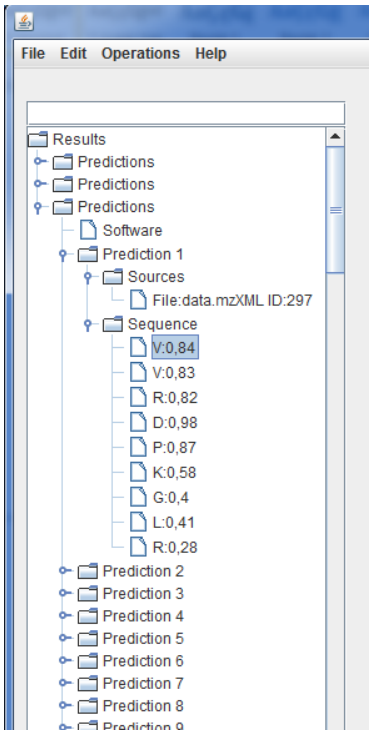

Figure 5.1. In this figure, it is seen that there is tree view for summarized representation about opening files. Opening file has many predictions and the third prediction is expanded to see information about sources which have one source and sequence which show amino acid and confidences.

At results in tree view, it shows some summarized opening files such as id which represent a number of spectrum in table, Spectra which show spectra data files, the size of the prediction which represents that the number of prediction is done in this spectrum file, Sequence which describe highest finding sequence in predictions of this spectrum file, confidence which represents confidence of the highest finding sequence in predictions of this spectrum file. Also, there is the search text area to find given text with threshold in table.

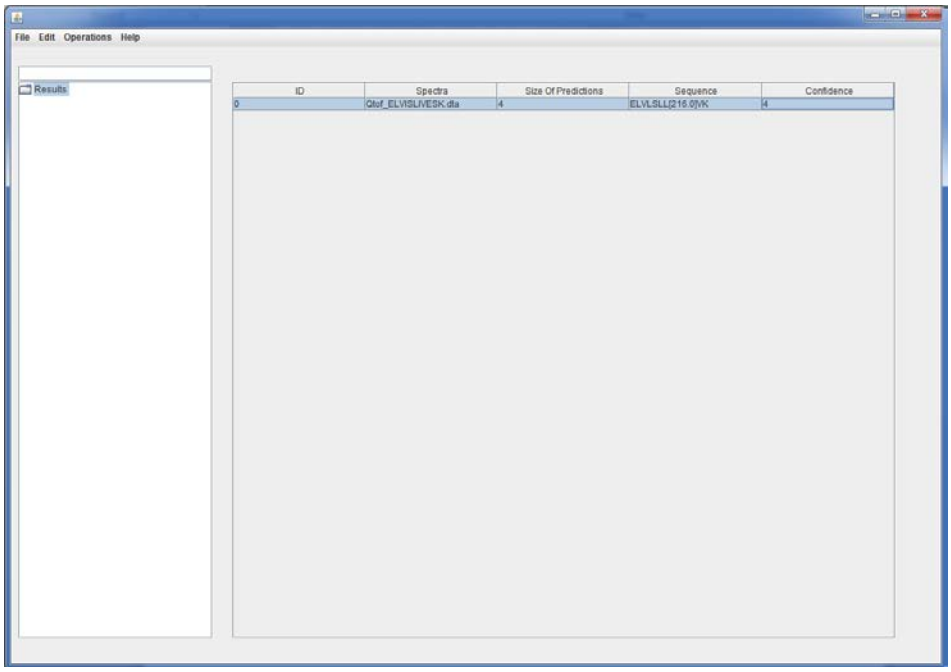

Figure 5.2. Figure show an overview of results section. It summarizes all information about opening files such as id which represent number of spectrum in table, Spectra which show spectra data files, size of prediction which represent that number of prediction is done in this spectrum file, Sequence which describe highest finding sequence in predictions of this spectrum file, confidence which represent confidence of highest finding sequence in predictions of this spectrum file.

If the results section in the tree view (Figure 5.1) is expanded, predictions are shown in that place. Predictions are a group of prediction which is predicted by the same software settings and algorithm. If Predictions button in tree view is clicked, summarized information of predictions such as id, Spectra, size of prediction, Sequence, confidence are shown in this software settings and algorithm.

Software node in tree view shows detailed setting about the software which is used for the predictions. Representation is shown in Figure 5.3.

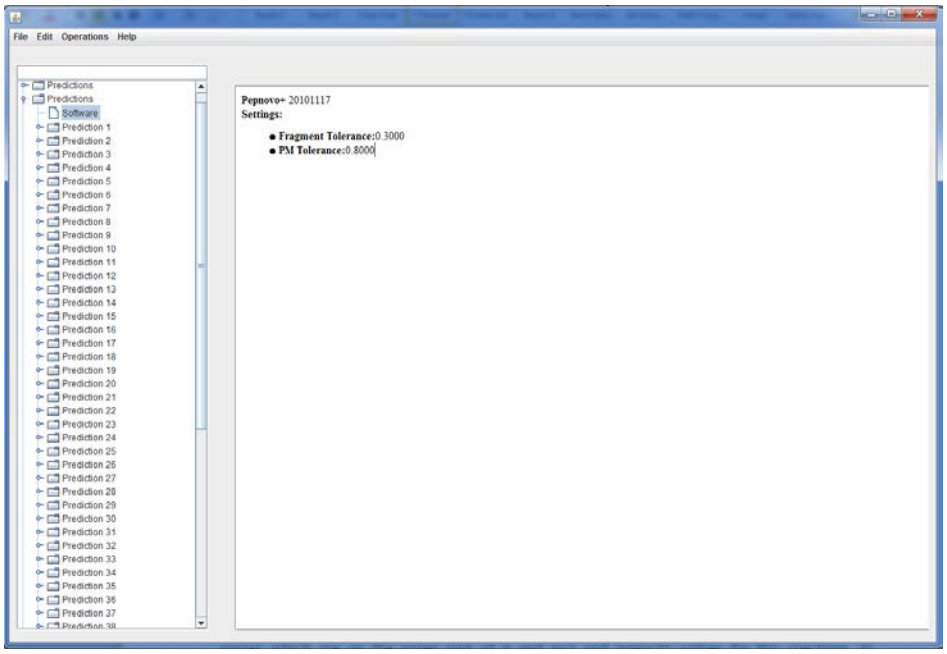

Figure 5.3. Software part shows all setting of it in clicked predictions. For example, software setting about pepNovo+ is seen in there.

When clicking prediction nodes in tree view, it is seen charts and information text area. In charts, all used spectra for this prediction is shown on the left part of program. Charts have file names which are on the upper part of it and m/z and intensity values for this spectrum. In information text area, information is given about this prediction such as the prediction number which is shown in bold at left of it, sources which is used for this prediction and listed at middle of it. What is more, assumed charge, scores, sequence, confidence, calculated mass are represented in that area. On right side, there is a table which show sequence elements and confidences of these sequence elements.

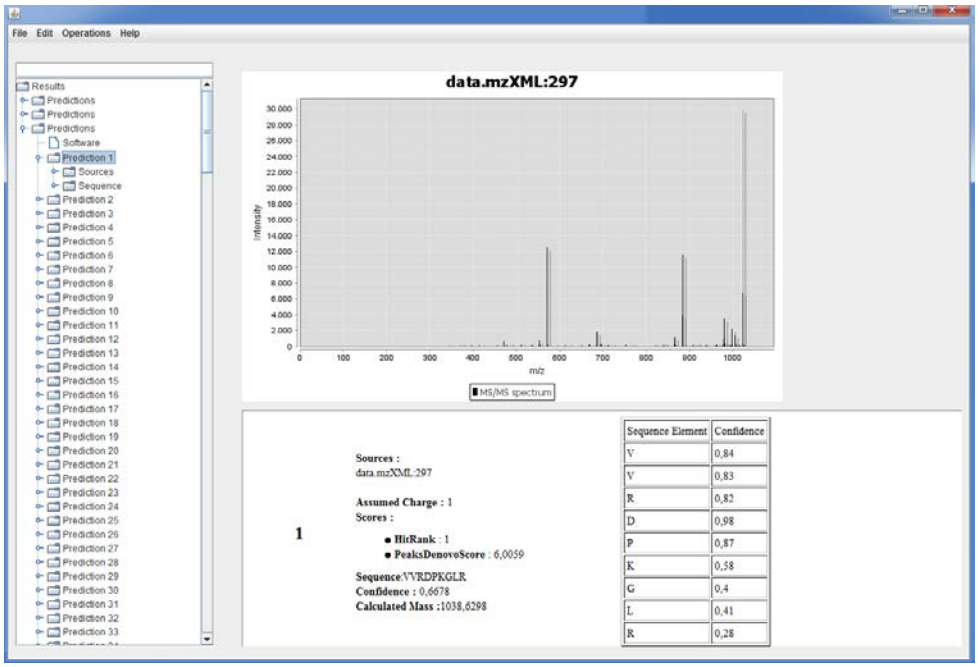

Figure 5.4. In the figure, prediction result is shown. In there, charts, sources and assumed charge about spectra of this prediction, finding sequence, confidence, confidence, calculated mass and scores of this sequence. What is more, there is a table which show sequence elements and confidence of these elements.

Sources node in tree view represents spectra of selected prediction. When clicking this node, spectra of selected prediction in chart are seen. What is more, there is text area that represents Precursor M/Z, Precursor Intensity In source node.

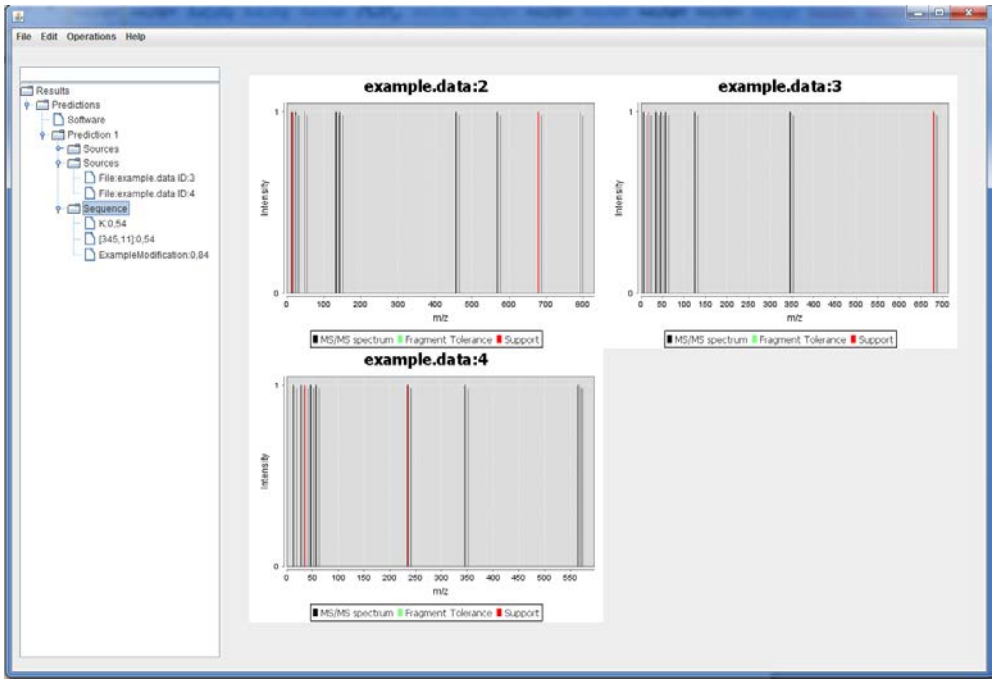

Figure 5.5. The Sequence is seen in this figure. In there, all spectra of selected prediction are represented by m/z and intensity values. What is more, all support elements is shown in each spectrum as red. In addition, fragment tolerances of support elements are represented with green.

Sequence node in tree view represents spectra for this sequence with all support. What is more, Sequence node has amino acid, modified amino acid and gap node. Each of them is represented by chart with fragment tolerance; support shows a location of this element in the spectrum and m/z, intensity values. It is seen in Figure 5.5.

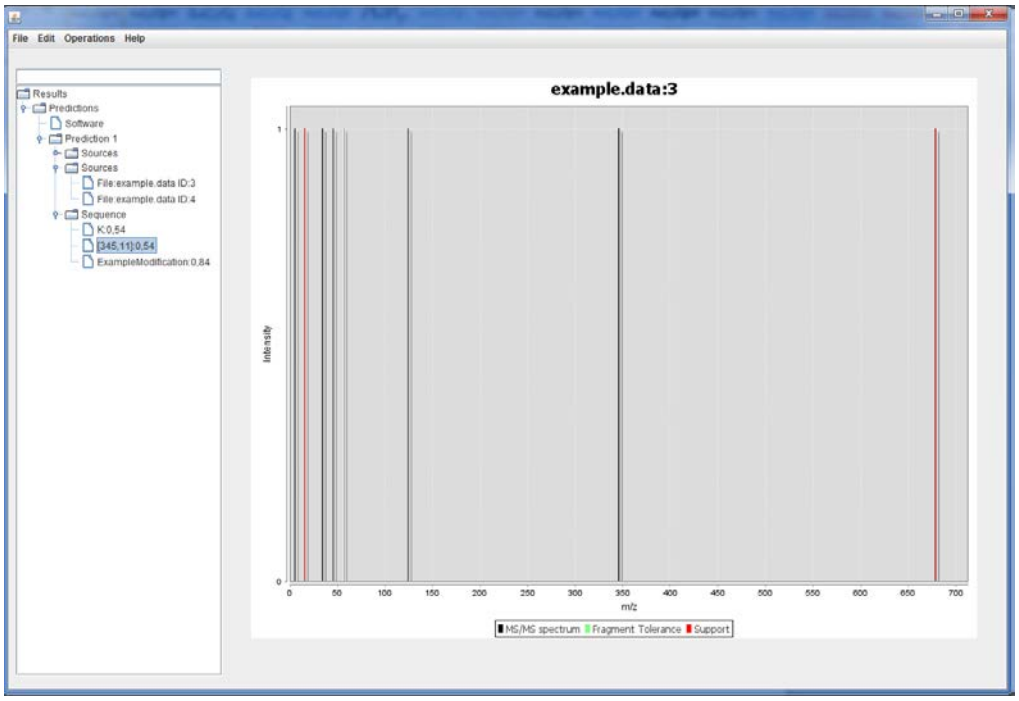

Figure 5.6. Each sequence element is shown in this figure. In this picture, one gap, amino acid and modified amino acid are represented. When clicking these elements, spectrum with m/z and intensity values, fragment tolerance and support are shown in chart.

CHAPTER 6

RESULT AND DISCUSSION

Currently, *de novo* prediction algorithms store all the information in their particular, if not peculiar, formats (Takan and Allmer 2012). Hence, algorithm-specific solutions are abundantly available. However, there is no a general solution to store and share *de novo* sequencing results which diminishes the usefulness of these results. Hence, they need to be integrated and further analyzed and cannot be used directly as evidence for peptide identification and protein sequencing at present. In order to make improvements the field of *de novo* sequencing a standard is vitally important.

Several areas of mass spectrometry-based proteomics have seen standardization such as for collections of mass spectra and MS/MS spectra in files like mzXML, mzData and most recently in mzML. The Human Proteome Organization (HUPO) oversees many standard initiatives and adapts community standards or actively develops new ones. Examples for community standards are protXML and pepXML which store database search algorithms' results. HUPO has published the mzIdentML standard to supersede pepXML and protXML.

As there is currently no standard to represent *de novo* sequencing results, others are not suitable formats used to store *de novo* predictions. PEAKS, commercial *de novo* sequencing tool, for instance, stores results in pepXML format. Other *de novo* sequencing algorithms such as PepNovo and Lutefisk, use custom formats which unfortunately do not provide adequate information for further downstream analyses and knowledge mining. It is necessary to develop a standard representation for *de novo* sequencing results since none of the current programs represent the minimum information needed and standards from related fields are either flawed or not fully applicable.

In *de novo* sequencing making a prediction based on multiple spectra is becoming more important. In the light of this information, a serious design limitation of the proposed pepXML format is that each result can only be linked to one spectrum source. PepXML and mzXML files can become humongous (sometimes several gigabytes). This does ease the exchange of data but moves the problem from a

standardization issue to a technical difficulty. Since as many as 90% of spectra may remain unassigned, the two options are viable to overcome the problem: one to link to specific spectra in an mzML or mzXML file and the other to include the spectrum in the exchange format. This solution leads to the possibility of exchanging only one file containing only relevant information. Automatic merging of documents, although not a vital feature is currently not supported for any of the standards. PepXML has been designed for the representation of the best result in mind. For *de novo* prediction this is a limitation since the search space is much larger than database search algorithms thus multiple equally well results are frequent and need to be represented alongside each other to enable further analyses. Although 100 post translational modifications are known, in current standards the user is expected to provide possible modifications. It is believed that a format to represent search results in a database like pepXML should model only that result. However, instead information is inflated by many results of in-house tools provided by the developers of the standard like XPRess, ASAPRatio and PeptideProphet. Interestingly, it would be possible to store the results of PeptideProphet, for example, in pepXML, without it being explicitly modeled, as it is now. Another issue along these lines is that a data model may never include visualization instructions, but the pepXML format includes such information as well as information which could easily be calculated from data modeled in the file and hence it need not be stored.

In an attempt to overcome all these limitations, the *de novo* markup language (DNML) and *De Novo* MS Ontology (DNMSO) are developed. The format proposed here is meant as a standard for the representation of *de novo* sequencing results. First and foremost, we ensured that all necessary, but as little unnecessary data as possible, is represented in the format. Another issue that it is identified to be crucial to the acceptance of a standard initiative is the availability of advanced programming interfaces (API) to read and write the standard. Many standards fail to offer an API and if they do only for reading of the standard. To increase the acceptance of DNML and DNMSO offer reading, writing, and conversion facilities. But most importantly, API allows for the creation of files and can thus be used directly in any software that performs *de novo* sequencing. This removes the need to develop reading and writing functionality for all developers of *de novo* sequencing algorithms, a mundane task which is often neglected and therefore often produces peculiar outcomes. Finally,

reading of mzXML and mzML information is included in the API so that any developer of *de novo* sequencing software can completely focus on the algorithm.

What is more, Graphical user interface has been created for visualization and analysis using this advanced programming interface. In this respect, predictions, spectra and relations between them are shown easily. Spectra are represented by charts .Also, all operations such as merging, adding, deleting, transferring data and converting can do with GUI. The laboratory scientists are the main users of the DNMSO Analyzer which does not require proteomics support to get started.

CHAPTER 7

CONCLUSION

Often corruption occurs while parsing files or files that don't not exactly adhere to their target standards. This is clearly undesirable and most likely due to the redundant development of APIs by different vendors. Therefore, an API is provided to remove this problem since all functionality, reading, creating, and writing are available thus removing the possibility of creating files that do not validate or are corrupted. This also enhances the confidence of developers of downstream tools, building on data provided in DNML and DNMSO format. The link is used to spectra in mzXML and mzML format which can be accessed by id or extracted as text, further enhances the ease with which API can be used. API thus facilitates all necessary input for *de novo* sequencing and all output thus completely abstracts file handling from the developers of *de novo* algorithms. Since many spectra have been processed by the existing *de novo* prediction tools and since those results are not available in DNML and DNMSO format conversion facilities have been implemented for the currently most prominent free tools such as LutefiskXP and PepNovo and for the commercial algorithm PEAKS.

The DNML schema, The DNMSO ontology and the fully fledged API are the main outcomes of this study. Especial attention is paid to the extensibility of the API and most parts of the code allows for runtime binding of additional functionality thus making it possible for developers to easily extend the API.

In order to allow the experimental proteomics community to analyze data stored in the DNMSO standard, Graphical User Interface is developed a, DNMSO Analyzer , an open source graphical user interface, to provide some facilities such as reading of various spectra file formats, reading, viewing, summarizing of DNMSO, and several conversions from existing *de novo* results to DNMSO in DNMSO Analyzer .

For testing all these, several dataset is used. First dataset is measured with the Applied Biosystems MDS SCIEX 4000 ESI Q-Trap and the Thermo Scientific LTQ XL Linear Ion Trap ESI mass spectrometers (MS) with CID fragmentation, respectively. MzML dataset was obtained from http://www.psidev.info/mzml/. In this project, LutefiskXP, PepNovo, Peaks which are *de novo* sequencing algorithms are used for

creating other data sets which are *de novo* results. What is more, API has been used in COMAS, another *de novo* sequencing algorithm created by JLAB (http://bioinformatics.iyte.edu.tr/) in Izmir institute of technology, in the visualization tool which was developed in this project for testing. All these *de novo* sequencing algorithms run with the same parameters. In this respects, these data sets converted to DNMSO files by API. Hence, functionality, effectively, usability of our DNMSO API is tested by writing, merging, reading, converting functions. Also, DNMSO manipulator is tested with all these data.

The first aim of this thesis was creating new standard that provide many to many relationships between spectra and predictions, exchange and merging, showing all results, PTMs, compact, no using proprietary formats and this was achieved as can be seen by DNML and DNMSO standards.

The second aim of this thesis was that an advanced programming interface which is standard, compact, modular, and easily extensible and also has read, writes, creates, converts, supports current standards facilities wants to be developed and it was achieved as exemplified by DNML and DNMSO API.

The third aim of this thesis was that Graphical user interface wants to be created for visualization, analysis, merging, adding, deleting, transferring data, converting, seeing spectra, predictions and relations between them and it was achieved as exemplified by DNMSO Analyzer .

REFERENCES

Aebersold R, Goodlett DR (2001) Mass Spectrometry in Proteomics. Chemical Reviews 101:269–296. doi: 10.1021/cr990076h

Aebersold R, Mann M (2003) Mass spectrometry-based proteomics. Nature 422:198–207. doi: 10.1038/nature01511

Allmer J (2011) Algorithms for the de novo sequencing of peptides from tandem mass spectra. Expert review of proteomics 8:645–57. doi: 10.1586/epr.11.54

Allmer J, Markert C, Stauber EJ, Hippler M (2004) A new approach that allows identification of intron-split peptides from mass spectrometric data in genomic databases. FEBS letters 562:202–6. doi: 10.1016/S0014-5793(04)00212-1

Alterovitz G, Xiang M, Hill DP, et al. (2010) Ontology engineering. Nature Biotechnology 28:128–30. doi: 10.1038/nbt0210-128

Baldwin MA (2004) Protein identification by mass spectrometry: issues to be considered. Molecular & cellular proteomics: MCP 3:1–9. doi: 10.1074/mcp.R300012-MCP200

Dancík V, Addona TA, Clauser KR, et al. De novo peptide sequencing via tandem mass spectrometry. Journal of computational biology: a journal of computational molecular cell biology 6:327–42. doi: 10.1089/106652799318300

Deutsch EW (2010) Mass spectrometer output file format mzML. Methods in molecular biology (Clifton, NJ) 604:319–31. doi: 10.1007/978-1-60761-444-9_22

Domon B, Aebersold R (2006) Mass spectrometry and protein analysis. Science (New York, NY) 312:212–7. doi: 10.1126/science.1124619

Downard K (2004) Mass Spectrometry. 210. doi: 10.1039/9781847551306

El-Aneed A, Cohen A, Banoub J (2009) Mass Spectrometry, Review of the Basics: Electrospray, MALDI, and Commonly Used Mass Analyzers. Applied Spectroscopy Reviews 44:210–230. doi: 10.1080/05704920902717872

Eng JK, McCormack AL, Yates JR (1994) An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database. Journal of the American Society for Mass Spectrometry 5:976–989. doi: 10.1016/1044-0305(94)80016-2

Eng JK, Searle BC, Clauser KR, Tabb DL (2011) A face in the crowd: recognizing peptides through database search. Molecular & cellular proteomics : MCP 10:R111.009522. doi: 10.1074/mcp.R111.009522

Figeys D (2005) Proteomics: the basic overview. Methods of biochemical analysis 45:1–62.

Frank A, Pevzner P (2005) PepNovo: De Novo Peptide Sequencing via Probabilistic Network Modeling. Analytical Chemistry 77:964–973. doi: 10.1021/ac048788h

Gruber TR (1993) Technical Report KSL 92-71 Revised April 1993 A Translation Approach to Portable Ontology Specifications by A Translation Approach to Portable Ontology Specifications. Knowledge Creation Diffusion Utilization 5:199–220.

Hirosawa M, Hoshida M, Ishikawa M, Toya T (1993) MASCOT: multiple alignment system for protein sequences based on three-way dynamic programming. Bioinformatics 9:161–167. doi: 10.1093/bioinformatics/9.2.161

Jones AR, Eisenacher M, Mayer G, et al. (2012) The mzIdentML data standard for mass spectrometry-based proteomics results. Molecular & cellular proteomics: MCP 11:M111.014381. doi: 10.1074/mcp.M111.014381

Keller A, Eng J, Zhang N, et al. (2005) A uniform proteomics MS/MS analysis platform utilizing open XML file formats. Molecular systems biology 1:2005.0017. doi: 10.1038/msb4100024

Liu H-L, Hsu J-P (2005) Recent developments in structural proteomics for protein structure determination. Proteomics 5:2056–68. doi: 10.1002/pmic.200401104

Ma B, Zhang K, Hendrie C, et al. (2003) PEAKS: powerful software for peptide de novo sequencing by tandem mass spectrometry. Rapid communications in mass spectrometry : RCM 17:2337–42. doi: 10.1002/rcm.1196

Martens L, Chambers M, Sturm M, et al. (2011) mzML--a community standard for mass spectrometry data. Molecular & cellular proteomics: MCP 10:R110.000133. doi: 10.1074/mcp.R110.000133

McGuinness D, Harmelen F Van (2004) OWL web ontology language overview. W3C recommendation 1–22.

Orchard S, Taylor C, Hermjakob H, et al. (2004) Current status of proteomic standards development. Expert review of proteomics 1:179–83. doi: 10.1586/14789450.1.2.179

Patterson SD, Aebersold RH (2003) Proteomics: the first decade and beyond. Nature genetics 33 Suppl:311–23. doi: 10.1038/ng1106

Pedrioli PGA, Eng JK, Hubley R, et al. (2004) A common open representation of mass spectrometry data and its application to proteomics research. Nature biotechnology 22:1459–66. doi: 10.1038/nbt1031

Seidler J, Zinn N, Boehm ME, Lehmann WD (2010) De novo sequencing of peptides by MS/MS. Proteomics 10:634–49. doi: 10.1002/pmic.200900459

Shadbolt N, Berners-Lee T, Hall W (2006) The Semantic Web Revisited. IEEE Intelligent Systems 21:96–101. doi: 10.1109/MIS.2006.62

Smith B, Ceusters W, Klagges B, et al. (2005) Relations in biomedical ontologies. Genome biology 6:R46. doi: 10.1186/gb-2005-6-5-r46

Steen H, Mann M (2004) The ABC’s (and XYZ's) of peptide sequencing. Nature reviews Molecular cell biology 5:699–711. doi: 10.1038/nrm1468

Takan S, Allmer J (2012) De novo markup language, a standard to represent de novo sequencing results from MS/MS data. 2012 7th International Symposium on Health Informatics and Bioinformatics. IEEE, pp 31–36

Taylor JA, Johnson RS (2001) Implementation and Uses of Automated de Novo Peptide Sequencing by Tandem Mass Spectrometry. Analytical Chemistry 73:2594–2604. doi: 10.1021/ac001196o

Wang X, Zhang D (2004) Ontology based context modeling and reasoning using OWL. Pervasive Computing and Communications Workshops, 2004. 18–22. doi: 10.1109/PERCOMW.2004.1276898

Steehler JK (2009) Introduction to Mass Spectrometry: Instrumentation, Applications, and Strategies for Data Interpretation, 4th Edition (by J. Throck Watson and O. David Sparkman). Journal of Chemical Education 86:810. doi: 10.1021/ed086p810.1

Wilde E, Glushko RJ (2008) XML fever. Communications of the ACM 51:40. doi: 10.1145/1364782.1364795

Ritke MK (2006) Essential Bioinformatics Jin Xiong Cambridge University Press; ISBN: 0-521-60082-0; 339pp.; 2006; Briefings in Bioinformatics 8:65–67.