Software Engineering 2012, 2(4): 147-159
DOI: 10.5923/j.se.20120204.08

Formal UML Modelling of Isotopo, Bioinformatical Software for Mass Isotopomers Distribution Analysis

Zeeshan Ahmed[*], **Saman Majeed** , **Thomas Dandekar**

Department of Bioinformatics, Biocenter, University of Wuerzburg, Germany

Abstract Mass isotopomer distribution analysis (MIDA) is a technique towards the measurement of amalgamation of polymers by involving the process of quantification of relative abundances of molecular species with mass spectrometry. The objective of this research is to study metabolic isotopes to quantify the fraction of metabolites of interest in the mixture typically by tracing isotopes. Estimating mass isotopomers distribution from spectral data is an extension of the quantitative mass spectrometric method to a multi component mixture analysis. Focusing on identifying the quantity of population of labelled isotopomers for resolving the exact rate of synthesized fractions present in the mixture and metabolic experimental data management, a new software application named "Isotopo" is proposed and designed. Isotopo is an application with proposed abilities of performing quantitative mass spectrometry to readily mixtures of materials labelled with stable isotopes. This can be very important for both biomedicine and biochemistry. Most recent version of Isotopo will have the ability of processing experimental isotopomers data and estimating mass values and relative intensities. Using formal mathematical algorithms which generate an appropriate set of linear simultaneous equations, it will predict natural abundance values, relative isotopic abundance values and fractional molar abundance values for each fragment from labelled substance based experimental data elements. Using Isotopo it will also possible to process data sets with multiple data entries up to three actual intensity values against one mass to charge ratio values, estimate absolute enrichment, mean and standard deviation of both natural and relative isotopic abundances. Isotopo will also provide the standardization of experimental data with a file based record keeping system for experimental data manipulation and management. In this paper justifying the need of a new software application, we also present the followed V-Model, formal UML designs (including use case, data flow, flow chart, system sequence, component and class diagrams), and designed graphical user interface of Isotopo.

Keywords Bioinformatics, Design, Human Computer Interaction (HCI), Mockup, Mass Isotopomers Distribution Analysis (MIDA), Metabolic Flux Analysis (MFA), Unified Modelling Language (UML)

1. Introduction

Bioinformatics is one of the recently introduced and highly contributing fields towards empirical, computational and complex data analysis with the involvement of probability, statistics, mathematics and informatics (e.g.[1]). It is providing heavy experimental data management and manipulation using relational database management system (e.g.[2],[3]) and globalizing data at web using grid[4] and semantic web[5] technologies etc. Bioinformatics has already challenged to explore several natural sciences areas e.g. metabolic network analysis and (re)construction, automation of genome annotation, protein structure determination etc., and provided values to the field of biology (& related) but still lots of areas need to be targeted and improved.

* Corresponding author: Zeeshan Ahmed
zeeshan.ahmed@uni-wuerzburg.de (Zeeshan Ahmed)
Published online at http://journal.sapub.org/se

The objective of this research is to study metabolic isotope to quantify the fraction of metabolites of interest in the mixture typically by tracing isotopes. Mass isotopomer distribution analysis (MIDA) measures the mixtures of polymers (e.g. *lipids, carbohydrates* and *proteins*) by quantifying relative abundances of molecular species with Mass Spectrometry (MS; a systematic technique to measure the mass-to-charge ratio values of charged particles)[6]. MS converts individual molecules into ions to direct them in magnetic fields using Mass Spectrometer[7]. During GC-MS, at first a mixture of compounds is inserted into the GC to vaporize using a heated chamber to separate compounds for MS analysis, by travelling into GC column.

A chromatogram is drawn, representing each compound with its peak. All Mass Spectrometers consists of three main sections: Ionizer, Ion Analyzer and Detector[8] (Figure 1). Electron impact ionization is performed by Ionizer with a gas chromatograph using a high-energy electron beam to collect molecular ions and fragments. Ion Analyzer accelerates obtained molecular ions and fragments by manoeuvring the charged particles using mass spectrometer, eliminating uncharged molecular ions and fragments. The job of the Detector is to generate an electronic signal at every ion hit. During this process mass analyser classifies the ions with respect to the mass to charge ratio values and detector extracts the abundance values of each mass to charge ratio value.

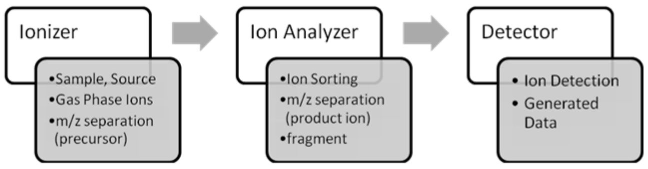


Figure Legend. GC-MS process consists of three main steps: Ionizer, Ion Analyzer and Detector, with supporting activities: sample source input, gas phase ions production and m/z separation (Ionizer), ion sorting, m/z separation and fragment (Ion Analyzer), and ion detection, generated data for further computation analysis (Detector)

Figure 1. Gas Chromatography Mass Spectrometry Process (GC-MS)

Estimating mass isotopomers distribution from spectral data is an extension of the quantitative mass spectrometric method to a multi component mixture analysis[9]. Mass spectrometric analysis describes relative abundances quantitatively based on combinatorial probabilities. It is essential to identify the quantity of the labelled isotopomer population to resolve the exact amount present in the mixture. The exact identification of the number of labelled isotopomers in a mixed population of molecules is a mathematical challenge. Different calculation algorithms have already been proposed and published for MIDA with overlapping solutions in successive iterations[10]. These generate identical results. A formal mathematical algorithm generates an appropriate set of linear simultaneous equations and finds their solutions, for the calculation of both natural isotope abundances and relative isotope abundances.

Some software applications do exists for mass isotopomers distribution analysis and metabolic modelling. As it is a broad field contributing towards the development of understanding between complex interactions, control and regulation of metabolic networks. Metabolic flux analysis is one of the key methods of metabolic modelling[11]. We (our group) have also developed some computational software applications (e.g. *Isotopo, Yana*[12], *Yanasquare*[12] and *Yanavergence*[13] etc.) for complex pathway analysis and isotopic distribution prediction[14]. Most importantly there is an existing solution provided by *Priv. Doz. Wolfgang Eisenreich's group at Institute of Biochemistry Technical University Munich Germany*, implementing the similar methodology but with completely different way of development and usage; in the form of a Microsoft excel macro. To process experimental metabolite data using developed macro, user at first needs to store experimental data in Microsoft excel sheets in a particular user unfriendly way and has to set many paths and configurations. To meet the aforementioned goals of the research, a new software application is needed to be designed and developed.

This manuscript presents research conducted to establish a user friendly platform for mass isotopomer distribution analysis (MIDA), another technique that enables the determination of metabolic fluxes on the basis of labelling experiments using 13C-enriched precursors. A new software application named 'Isotopo' is proposed and designed with facile data management and robustness to quantify the populations of isotopomers in mixtures of 13C-labelled amino acids. Isotopo will be the upgraded version of an existing software solution "*LS-MIDA*" towards MIDA[15], proposed and developed at *Prof. Dr. Thomas Dandekar's group of Functional genomics and systems Biology, Department of Bioinformatics, Biocenter at the University of Wuerzburg Germany*.

The proposed prototype will be with the ability of processing experimental isotopomer data and analysing quantitative mass spectrometry for isotopologue mixtures of compounds (e.g. amino acids) to derive metabolic fluxes.

The focus of software development will be towards the prediction of relative intensities with respect to the used mass to charge ratio values, natural abundances, relative abundances and fractional molar abundances of each fragment derived from the compound under study. To meet these goals, it will implement different mathematical algorithms including *least square*[16], *binomial theorem*[17] *matrix* (inverse, transpose etc.)[18][19], m*ean*, s*tandard deviation*, *linear*[20] *and multiple regression analysis*[21].

The novelty which will make this application unique of its own kind will be the processing of multi-intensity values against one mass to charge ratio value for absolute enrichments estimation of both natural and relative abundances for the underlying isotopologue, with repeated mathematical analysis for one experimental dataset. Furthermore, proposed prototype will be able to draw the spectrometry analysis to examine each peak of spectrum of given mass because if it is possible to find contributions of each molecular species, contribution from one heteroatom (with more than one abundant isotope) can also be sorted out[16], as the fragmentation of molecules containing heteroatom are difficult to understand because large number of fragments results from isotopic composition.

This manuscript mainly presents designed meta-models for Isotopo architecture as Bioinformatical Software for Mass Isotopomers Distribution Analysis. The overall scope taken is deliberately broad, aiming at covering multiple perspectives of a computational, empirical and data management based software product development. We are designing product line architecture[22] to make software application flexible enough to easily adopt future updates and additional features.

The manuscript is organized as follows: going from a more general overview, section 2 presents Isotopo V-Model (software development processes) and section 3 describes Isotopo UML[23][24] designs including *Use Case* (Section 3.1), *Data Flow* (Section 3.2), *System Sequence* (Section 3.3), *Internal Work Flow* (Section 3.4), *Component* (Section 3.5) and *Class* (Section .6) Diagrams. Section 4 gives the details of designed graphical user interfaces and Section 5 concludes the manuscript.

2. Isotopo: V-Model

The software development of any kind should be done following some process model. There are already some well established development models existing and followed e.g. *Waterfall model, Spiral model, Iterative and incremental development, Agile development, Code and fix,* and *some Process improvement models*. During our software design and development, we are following a well established software development model i.e. *V-Model*; an extended form of waterfall model proposed by Paul Rook[25].

The V-Model expresses the relationships between each phase of the development life cycle forming typical V shape[26] (Figure 2). The overall job of Isotopo V-Model software development process starts with the initialization of main concept (which in our case was MIDA), then scientific requirements for operational scenarios have to be clearly described and to be strictly followed to model Isotopo. Later on following architected software designs, a real time system has to be developed using programming, which then has to be in house tested (integrated) and validated by scientists. The final step is to maintain Isotopo and if needed then repeat V-Model for software releases with more computational and feature updates.

3. Isotopo: UML Modelling

As computational and empirical software systems development becomes more complex, scientific academias as well as commercial organizations require high-quality products in short time. Unfortunately usually wrong presumptions leads to direct software development without adopting software development life cycle and formal design modelling which gives a temporary and limited (scripted) solution and in the long run it is quite difficult to enhance and improve it. Software design modelling helps in dealing with complexity as the meta-model architecture provides abstraction and modification techniques which allows the designer to concentrate on the basis of a problem by reducing gratuitous details.

Today, a better way of architecture modelling for a newly proposed software application is available in the form of Unified Modelling Language (UML). It is a modelling language, a well suited and the standard way of designing software application by creating different abstract models. UML is capable of facilitating software engineers stand alone and interconnected semiformal (Meta) design views for modelling software architectures[23].

Here software designs are created using UML principles to have better understanding of Isotopo in terms of its implementation, usage and working, Designed UML diagrams describe over all feature based functionality, user accessibility, experimental data flow, internal system work flow, system sequence, involved component's integration and source code structure. In this manuscript we present following Isotopo UML diagrams: *Use Case, Data Flow, System Sequence, Internal Work Flow, Component* and *Class Diagrams*. This logical design presentation will give an overall physical view of the Isotopo focusing on its technical architecture, grouped functionalities, flow of information, operational perspective focusing on interface requirements and involved technologies during software design, development, deployment and testing.

3.1. Use Case

Use case is the specific textual and visual method of presenting software application's functionalities comprising all ways of user system interactions[27]. It consists of two main symbolic notations: *Actor* and *Activities*. In most of the cases actor is either user or system itself as a remote actor. Activity is the event triggered by the system in response to the request by actor for some action.

We have designed a use case diagram (Figure 3) and explained in detail (Table 1). The designed use case diagram describes the user system communication for the isotopomers experimental data analysis, which consists of a user (actor), five direct activities (*Execute LS, Input Data, Analyze data, Visualize Results, Save results*) and four indirect activities (*Open Data File, Enter data, Calculate natural abundance, Calculate relative*).

Use case diagram explains over all user system interaction. At first user needs to execute the software application Isotopo, then user can input experimental data in two ways to the software application for analysis, by entering manually and by loading experimental data file. After data inout, user can analyse it to calculate relative abundances. User can visualize obtained results by drawing a mass spectrum and also can save results in the form of an image file.

3.2. Data Flow Diagram (DFD)

Data Flow Diagram[28] presents the basic data flow inside the Isotopo Data Analyzer (Figure 4). Data has to be loaded from the Data File as input so called I/O Data, which will be then analysed by the system. Systematic analysis procedure is divided in to two levels.

First level starts by calculating relative and natural abundances using actual intensities using user inputted experimental. Then fractional molar abundance values and minimum abundance values are calculated using already calculated relative and natural abundance values.

In second level, again, new relative natural abundance values are calculated using previously calculated relative abundance values in level 1. Then relative abundance values are calculated using standard intensity values, inputted by user. Using these two newly calculated relative and natural abundance values, likewise level 1, fractional molar abundance and minimum values are calculated. In third level relative difference between the observations of level 1 and 2 is calculated.

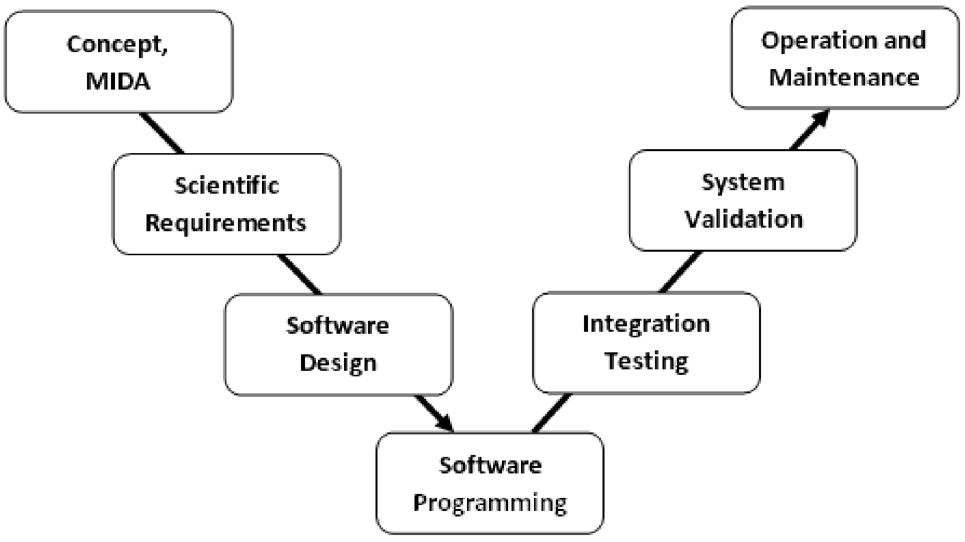


Figure Legend. Software development model consisting of seven phases: Concept MIDA, Scientific Requirements, Software Design, Software Programming, Integration Testing, System Validation and Operation and Maintenance.

Figure 2. Isotopo; V-Model Software Development Process

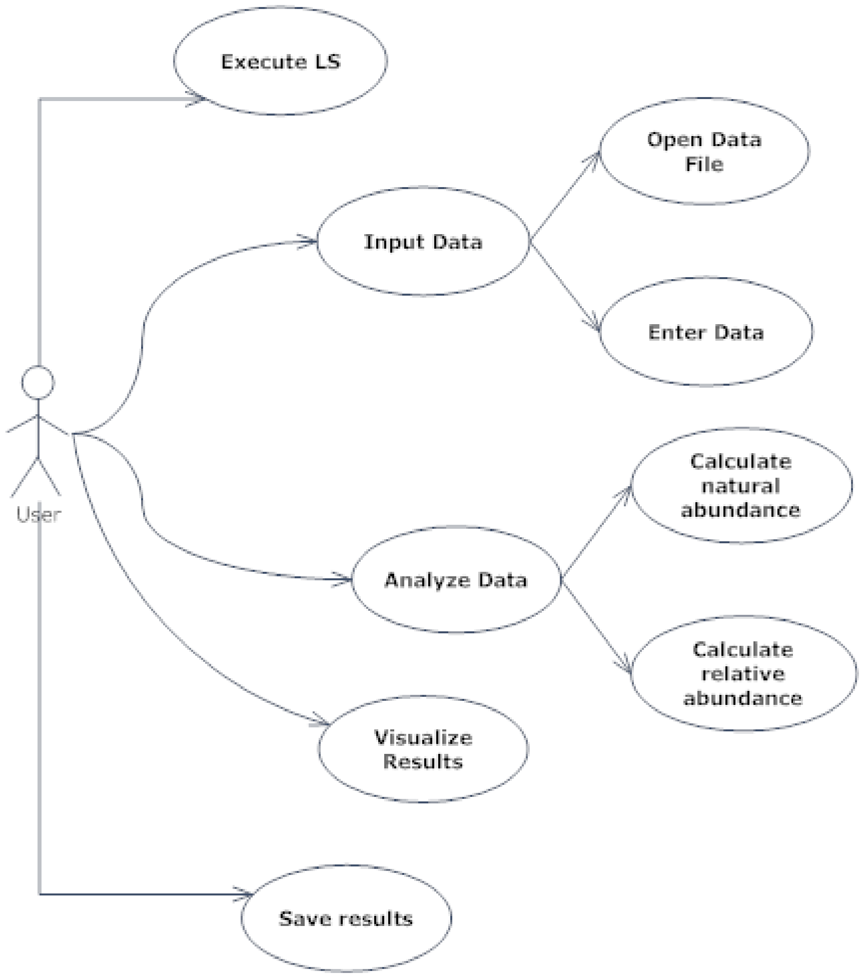


Figure Legend. Use case diagram of Isotopo is consisting of a User, five direct and four remote (indirect) activities.

Figure 3. Isotopo; Use Case

Table 1. Isotopo; Use Case

No.	Isotopo	
	Features	Descriptions
1	Number	1
2	Name	Isotopo Data Analyzer
3	Application	Isotopo
4	Description	This use case consists of a User, five direct and four remote (indirect) activities. This describes the user (actor) system (Isotopo) communication for the isotopomers experimental data analysis.
5	Primary Actor	User (1 Actor)
6	Precondition	Software application successfully running.
7	Trigger / Events	• Execute LS • Input Data • Open Data File • Enter data • Analyse data • Calculate natural abundance • Calculate relative • Visualize Results • Save results
8	Basic Flow	Basic flow consists of following steps: 1. Start software application 2. Enter input data by either loading from data file or by manually entering. 3. Analyse input data. 4. Visualize results 5. Observe predicted results (text and image). 6. Save obtained results for reuse.
9	Alternate Flows	Exception will be notified to the user.

3.4. System Sequence Diagram (SSD)

The System Sequence Diagram represents a particular scenario (text or graphic) defined by use case, especially for transaction oriented systems[29]. A SSD consists of actors (*users*), messages (*methods*) called by the actors, return values (optional, if any) and loop indicators. The main reason of using SSD is to explore the logic of multifaceted operations (procedures or functions).

The system sequence of Isotopo Analyzer is consists of seven sequential steps with individual tasks (Figure 5). At first the experimental data (*Metabolite, Actual Mass to charge ratio (M/Z) Values, Actual Relative Intensity (RI) Values, Standard M/Z Values, Standard RI Values and Number of Fragments*) has to be inputted to the system via the user via graphical user interface.

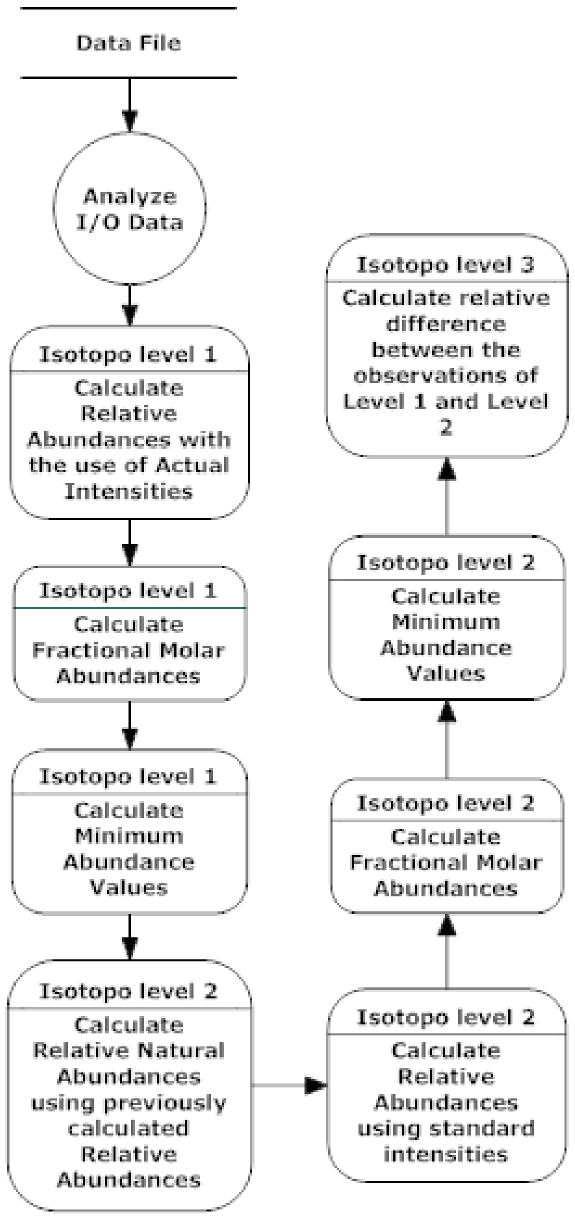

Data File
Analyze I/O Data
Isotopo level 1
Calculate Relative Abundances with the use of Actual Intensities
Isotopo level 1
Calculate Fractional Molar Abundances
Isotopo level 1
Calculate Minimum Abundance Values
Isotopo level 2
Calculate Relative Natural Abundances using previously calculated Relative Abundances
Isotopo level 2
Calculate Relative Abundances using standard intensities
Isotopo level 2
Calculate Fractional Molar Abundances
Isotopo level 2
Calculate Minimum Abundance Values
Isotopo level 3
Calculate relative difference between the observations of Level 1 and Level 2

Figure Legend. The data flow diagram of Isotopo is consisting of a File (Data File), one main Function (Analyze I/O Data) and eight internal functions: Calculate Natural Abundance Values and Calculate Relative Abundance Values, Calculate Fractional Molar Abundance, Calculate Minimum Abundance Values, Calculate Relative Natural Abundances using previously calculated Relative Abundances, Calculate relative difference between the observations of Level 1 and Level 2, Calculate new Minimum Abundance, Calculate new Fractional Molar Abundance and Calculate Relative Abundances using standard intensities

Figure 4. Isotopo; Data Flow Diagram

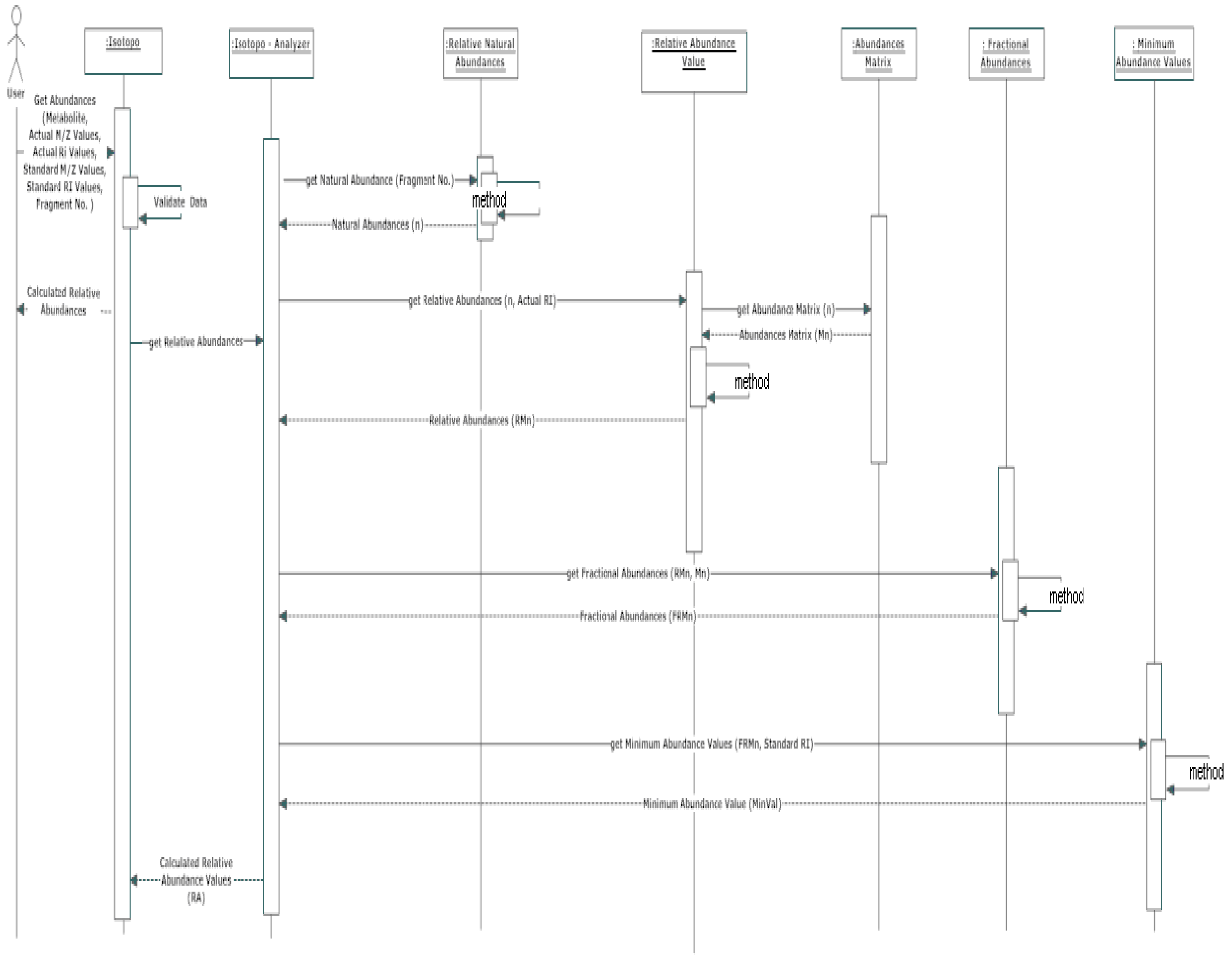

:Isotopo
:Isotopo - Analyzer
:Relative Natural Abundances
:Relative Abundance Value
:Abundances Matrix
: Fractional Abundances
: Minimum Abundance Values
User
Get Abundances (Metabolite, Actual M/Z Values, Actual Ri Values, Standard M/Z Values, Standard RI Values, Fragment No.)
Validate Data
get Natural Abundance (Fragment No.)
method
Natural Abundances (n)
Calculated Relative Abundances
get Relative Abundances (n, Actual RI)
get Abundance Matrix (n)
Abundances Matrix (Mn)
get Relative Abundances
method
Relative Abundances (RMn)
get Fractional Abundances (RMn, Mn)
method
Fractional Abundances (FRMn)
get Minimum Abundance Values (FRMn, Standard RI)
method
Minimum Abundance Value (MinVal)
Calculated Relative Abundance Values (RA)

Figure Legend. The abstract system sequence diagram of Isotopo is consisting of seven steps (Isotopo, Analyzer, Relative Natural Abundances, Relative Abundance Values, Abundance Matrix, Fractional Abundances and Minimum Abundance values) with several directing arrows in between.

Figure 5. UML System Sequence Diagram (SSD)

System at first analyses inputted data by validating it. After successful validation data will be sent to Isotopo Analyzer to calculate abundance values. Isotopo Analyzer then sends information based on number of fragments to the Relative Natural Abundances, which calculates relative natural abundances (n) and sends to Isotopo Analyzer.

Then, to calculate relative abundance values, Isotopo Analyzer sends calculated natural abundance values and actual Ri values to the step 4 i.e. Relative Abundance Values Ccalculator, which creates abundance matrix of calculated natural abundance values (Mn) using step 5 of system sequence i.e. Abundance Martix. Later after performing mathematical operations, sends back the resultant relative abundance values (RMn) to the Isotopo Analyzer.

Then, to calculate Fractional Molar Abundance values, Isotopo Analyzer sends natural abundance matrix values and calculated relative abundance values to the step 6 i.e. Fractional Abundances, which later after the mathematical operation performance sends back resultant fractional molar abundance values (FRMn) to the Isotopo Analyzer.

Later after, to calculate minimum abundance values, Isotopo Analyzer sends calculated fractional molar abundance values, standard relative intensity values to step 7 i.e., Minimum Abundance Values, which then later after the calculation of minimum values send back resultant values (Min Val) to the Isotopo Analyzer. This was the first complete transaction of different abundance value calculations.

3.3. Internal Work Flow Diagram

Internal flow chart is also known as the Flow chart; a step by step visual representation of defined interlinked processes (operations) in a software application[30], categorized in different shaped boxes representing different kinds of operations connected by directional and unidirectional (associated) arrows.

As the whole software application is divided in to two main modules: *Data Analyzer* and *Data Manager*. The internal work flow of Isotopo Data Analyzer (Figure6) starts with experimental data input from data file which then formatted by the system and resultant structured data will be displayed in graphical user interface of Isotopo Data Analyzer.

Later selected data by the user from graphical user interface is taken by the system to calculate natural, relative, fractional molar abundances and minimum values with the ratio of two. Then differences between to transitional abundance values is calculated and based on the resultant information a spectrum will be drawn by the system, which will be presented in graphical form at graphical user interface for the user visualization and analysis.

The internal work flow of the Isotopo data manager starts with experimental data manipulation (Figure 7), which leads to the experimental data extraction from existing data files and storing data by creating new data files. Furthermore it displays data loaded from data file into system and let user manipulate it by adding some new data, merging data from other files, deleting some data, and updating data.

3.5. Component Diagram

Component diagram is the visual presentation of assembled constituents representing structural relationship between service provider and consumer[24].

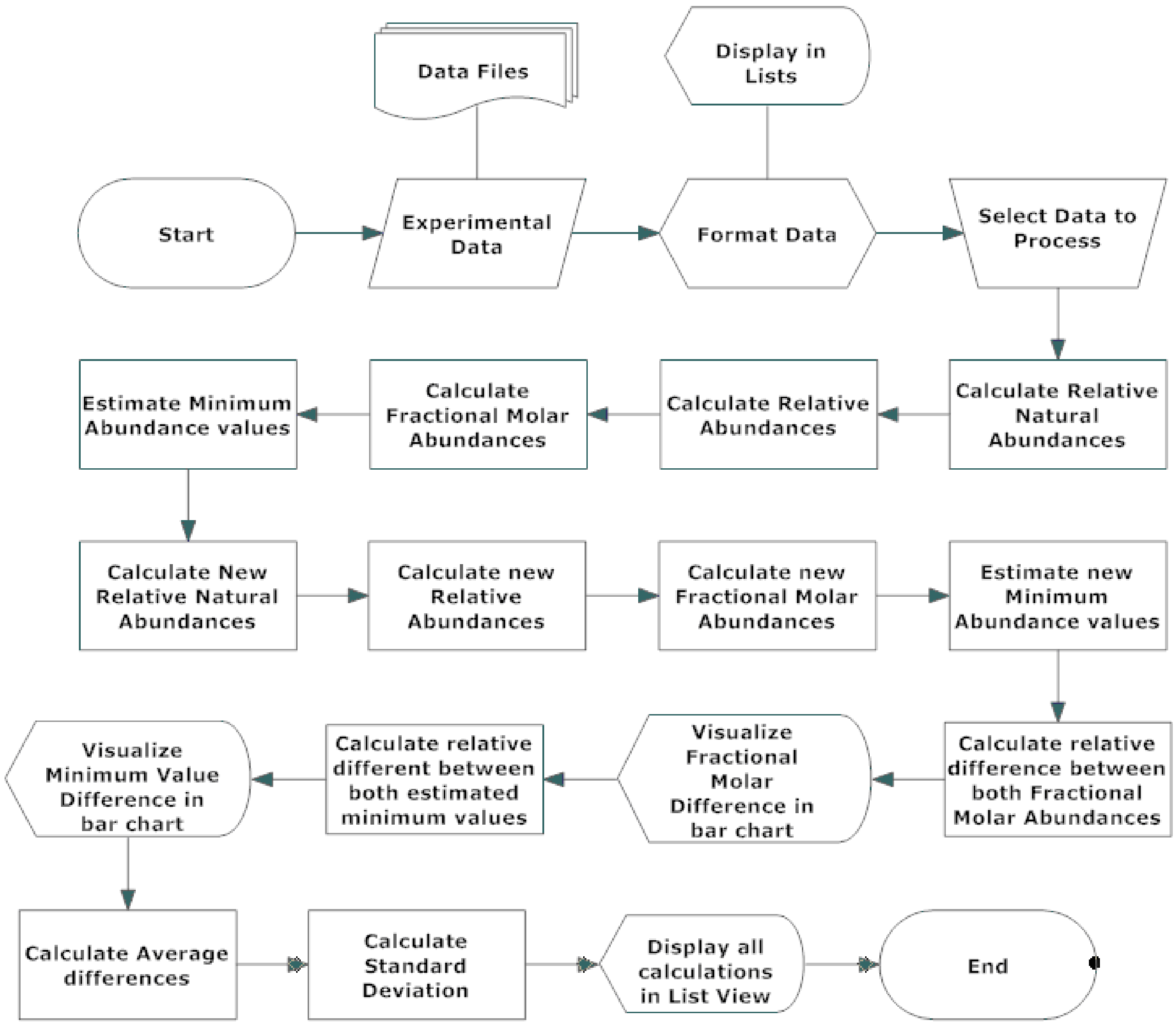

Data Files
Display in Lists
Start
Experimental Data
Format Data
Select Data to Process
Estimate Minimum Abundance values
Calculate Fractional Molar Abundances
Calculate Relative Abundances
Calculate Relative Natural Abundances
Calculate New Relative Natural Abundances
Calculate new Relative Abundances
Calculate new Fractional Molar Abundances
Estimate new Minimum Abundance values
Visualize Minimum Value Difference in bar chart
Calculate relative different between both estimated minimum values
Visualize Fractional Molar Difference in bar chart
Calculate relative difference between both Fractional Molar Abundances
Calculate Average differences
Calculate Standard Deviation
Display all calculations in List View
End

Figure Legend. . The flow chart of Isotopo Data Analyzer is consisting of one starting point (Start), one Input point (Experimental Data; Data Files), one formatting point (Format Data; Display in lists), one data selection point (Select Data to Process), processing units (Calculate Natural Abundance Values, Relative Abundance Values, Fractional Molar Abundance, Minimum Abundance Values, Average Differences and Standard Deviation), one visualization mode (Draw Spectrum) and one ending point (End).

Figure 6. Isotopo; UML Flow chart of Data Analyzer

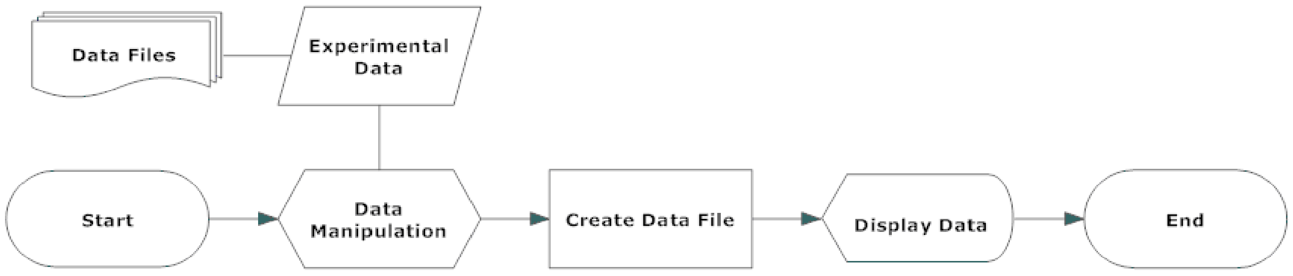


Figure Legend. The flow chart of Isotopo Data Manager is consisting of one starting point (Start), one Input point (Experimental Data; Data Files), one formatting point (Format Data; Display in lists), one data file creation process, one visualization mode and one ending point (End).

Figure 7. Isotopo; UML Flow chart of Data Manager

It allows the designer to confirm system's functionality to be implemented in the form of components using internal and third party services (e.g. programming languages, libraries, executables, application programming interfaces, frameworks etc.) in terms of nature and behaviour.

Isotopo is mainly consists of two modules i.e. Isotopo Data Analyzer and Isotopo Data Manager (Figure 8). User can access these both modules to perform mass isotopomers distribution estimation and file based experimental data management. Isotopo, as whole application, is developed using an object oriented platform independent language C Sharp (C#) and Microsoft Dot Net technology.

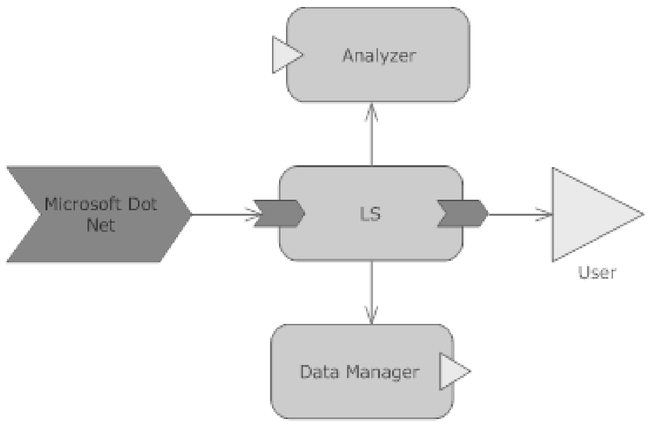


Figure Legend. The component diagram of Isotopo is consisting of one platform component (Microsoft Dot Net), one main component (LS), two subcomponents (Analyzer and Data Manager).

Figure 8. Isotopo; UML Component Diagram

3.6. Class Diagram

Class diagram is the static representation of relationships between defined classes for the development of a software application[31],[32]. The source code of Isotopo will be divided into three namespaces i.e. *SBEDA, System* and *ZEDGraph*. SBEDA is the main namespace containing all related and newly developed source code classes, System is the by default namespace provided by C-Sharp language used during the software development, and this namespace is responsible for providing access to default language based controls and components. Namespace REDGraph is a third party application programming interface used mainly for the development of graphical visualization of statistical, mathematical and experimental data in the form of two and three dimensional colored bar charts.

There are seven newly developed interlinked classes: *Main, IsotopoDataAnalyzer, Isotopo DataManager, Isotopo About, Calculation, Complex* and *Matrix* (Figure 9). As Isotopo is a multi document interface (MDI) application, Main MDI parent class which contains all other child classes. IsotopoDataAnalyzer is the multi attribute class developed as the graphical user interface of the Isotopo analyzer which provides all visual options to the user to load, edit, analyze and visualize experimental data and observed results.

IsotopoDataManager is the multi attribute class developed as the graphical user interface of the Isotopo Data Manager which provides all visual options to the user for file based experimental data management and manipulation including entering, loading, editing, updating, deleting, merging, replacing and saving data in files. Calculations is the multi attribute class developed for performing all mathematical operations including mass value estimations, relative abundances, data parsing and different data format conversions.

Matrix is the multi attribute class developed for performing matrix operations including drawing simple matrix of NxM rows and columns, calculation inverse and transpose of matrix. Complex is the multi attribute class developed for difficult mathematical operations including square root, absolute, tangent and operator overloading. IsotopoAbout is the single attribute class, providing information Isotopo and development team and research group.

Main sequence of classes, starts with Main container class, which provides other graphical user interface based classes IsotopoDataAnalyzer, IsotopoDataManager and IsotopoDataAbout. IsotopoDataAnalyzer perform user system communication, let user enter, edit and visualize experimental data, and analyze experimental data by directly using class Calculations which the uses classes i.e. Matrix and Complex. IsotopoDataManager is an independent multi attribute class performing operations including user system communication for file based data management and manipulations.

LeastSquareDataManager is the multi attribute class developed as the graphical user interface of the Isotopo

Data Manager which provides all visual options to the user for file based experimental data management and manipulation including entering, loading, editing, updating, deleting, merging, replacing and saving data in files. Calculation is the multi attribute class developed for performing all mathematical operations including mass value estimations, relative abundances, data parsing and different data format conversions.

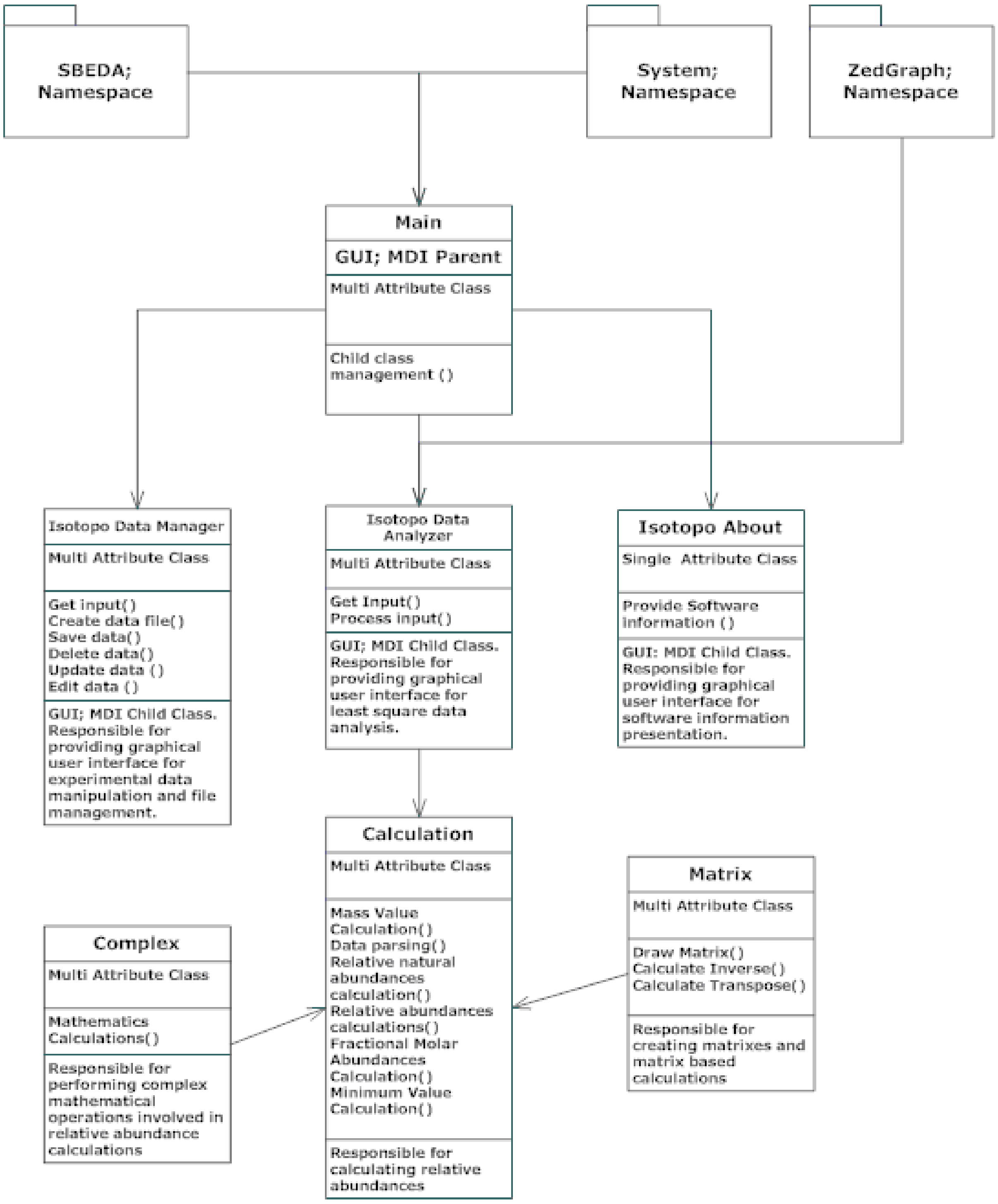
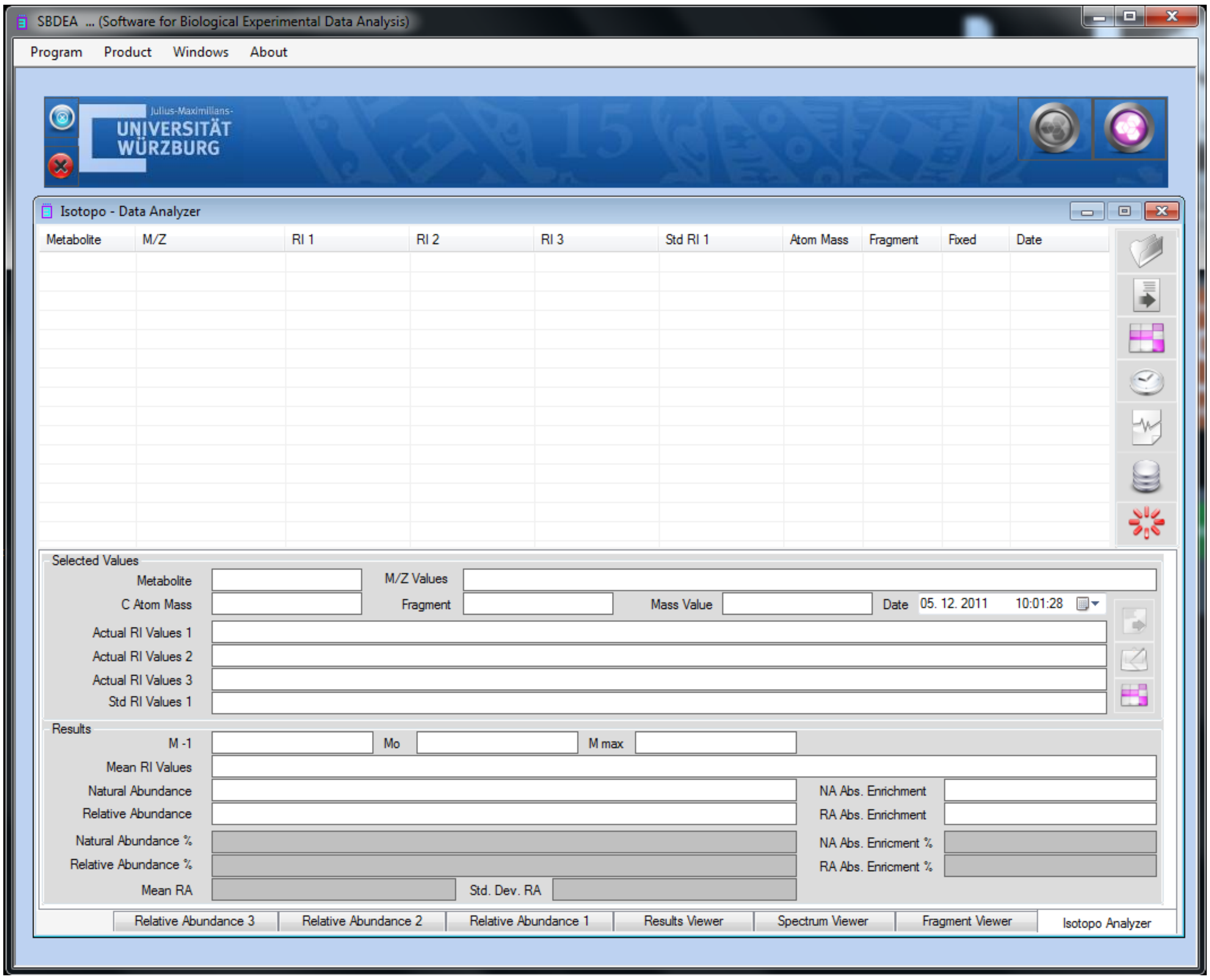

SBEDA; Namespace
System; Namespace
ZedGraph; Namespace
Main
GUI; MDI Parent
Multi Attribute Class
Child class management ()
Isotopo Data Manager
Multi Attribute Class
Get input()
Create data file()
Save data()
Delete data()
Update data ()
Edit data ()
GUI; MDI Child Class. Responsible for providing graphical user interface for experimental data manipulation and file management.
Isotopo Data Analyzer
Multi Attribute Class
Get Input()
Process input()
GUI; MDI Child Class. Responsible for providing graphical user interface for least square data analysis.
Isotopo About
Single Attribute Class
Provide Software information ()
GUI: MDI Child Class. Responsible for providing graphical user interface for software information presentation.
Calculation
Multi Attribute Class
Mass Value Calculation()
Data parsing()
Relative natural abundances calculation()
Relative abundances calculations()
Fractional Molar Abundances Calculation()
Minimum Value Calculation()
Responsible for calculating relative abundances
Complex
Multi Attribute Class
Mathematics Calculations()
Responsible for performing complex mathematical operations involved in relative abundance calculations
Matrix
Multi Attribute Class
Draw Matrix()
Calculate Inverse()
Calculate Transpose()
Responsible for creating matrixes and matrix based calculations

Figure Legend. The class diagram of Isotopo consisting of three name spaces (SBEDA, System and ZEDGraph), one main class (Main), six multi attribute classes (IsotopoDataAnalyzer, IsotopoDataManager, IsotopoAbout, Calculation, Complex and Matrix) and one single attribute class (IsotopoAbout).

Figure 9. Isotopo; UML Class Diagram



Date 05. 12. 2011 10:01:28

Figure 10 (a). Designed Main Graphical User Interface

Matrix is the multi attribute class developed for performing matrix operations including drawing simple matrix of NxM rows and columns, calculation inverse and transpose of matrix. Complex is the multi attribute class developed for difficult mathematical operations including square root, absolute, tangent and operator overloading. LeastSquareAbout is the single attribute class, providing information on Isotopo and development team and research group.

4. GUI Design

This section intends to provide an overview of Graphical User Interface (GUI) designing and basic understanding of the proposed Isotopo GUI. Targeting the challenge of the proposition of designing a standardized graphical user interface, a review research is conducted in a chosen the field i.e. Human Computer Interaction (HCI), to have complete understanding of graphical user interface design and development. HCI is renowned as Human Machine Interface (HMI); the study of designing, evaluating and implementing interactive computing systems for human use[33]. Designing High quality HCI design is difficult to implement because of many reasons: market pressure of less time development, rapid functionality addition during development, excessive several iterations, competitive general purpose software and human behaviour analysis.

Designing human computer interaction interface is an important and a complex task, but it could be simplified by decomposing task into subcomponents and maintaining relationships among those subcomponents. Task decomposition is a structured approach, applicable in both Software Engineering and Human Computer Interaction (HCI) fields depending on specific processes and design artifacts. Using design artifacts applications could be made for analysis and design by making the hand draw sketches to provide high level of logical design based on user requirements, usage scenarios and essential use cases. To design hand drawn sketches there are some strategies to be followed .i.e., planning, sequential work flow, and levels of details. While evaluating or designing a user interface, it is important to keep in mind the HCI design principles. There are four major HCI design principles .i.e., Cooperation, Experimentation, Contextualization, Iteration and Empirical Measurement[33].

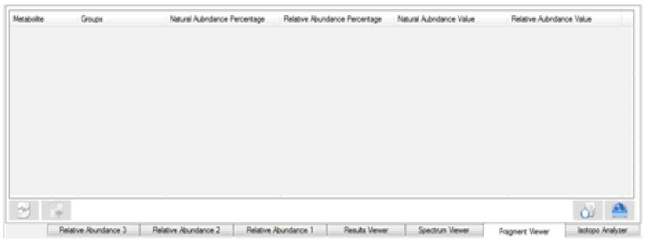
Like software engineering design patterns there are some graphical user interface design patterns (Window Per Task, Direct Manipulation, Conversational Text, Selection, Form, Limited Selection Size, Ephemeral Feedback, Disabled Irrelevant Things, Supplementary Window and Step-by-Step Instructions) needs to be strictly followed in software graphical user interface design. These patterns help designers in analysing already designed graphical interfaces and designing a user friendly and required on demand graphical interface.

As the proposed software will probably be used by the scientist belonging to the fields of Biology and related fields (e.g. Biochemistry etc.), the major need of the design is to be simple and easy to use, as most of the time people belonging to aforementioned fields are not interested in learning and spending time familiarizing their selves to new software application.

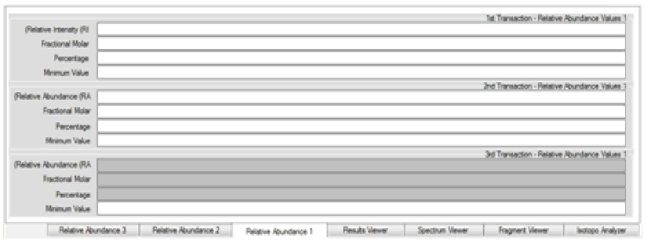
The designed graphical user interface[34] of Isotopo Data Analyzer is presented in Figure 10 and Data Manager in Figure 11. The graphical user interface of Isotopo data analyzer consists of 10 main controls: open data file, clear all text controls, measure selected data, process all data, remove selected data, open data manager, close Isotopo, selected values and results.

Moreover the graphical interface is divided into seven views: Isotopo Analyzer, Fragment Viewer, Spectrum Viewer, Result Viewer, Relative Abundance 1, Relative Abundance 2 and Relative Abundance 3.

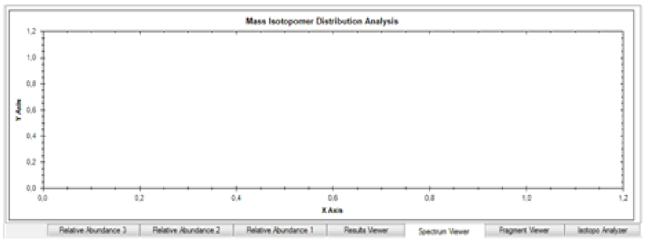
The graphical user interface of Isotopo Data Manager consists of 16 main controls: open data file, clear all text controls, close isotopo data manager, add new values, update edited values, clear text fields, save data in file, select values to edit, delete values, create new data file, select source directory, save file, cancel creating file, data view, Open Isotopo Data Analyzer and Open Isotopo Data Viewer.



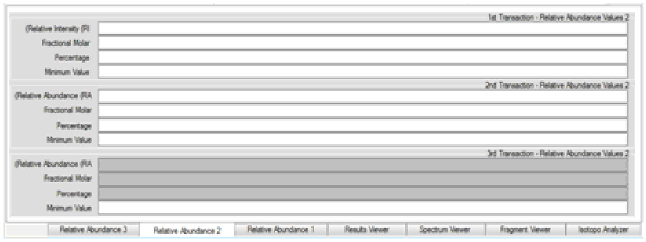
(b): Isotopo; Fragment Viewer



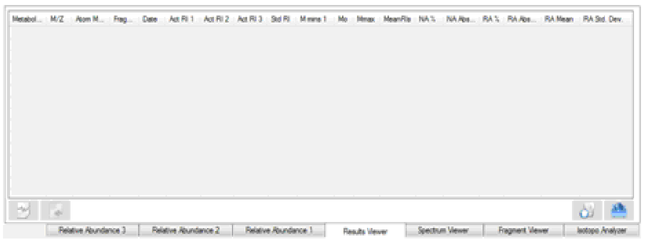
(e): Isotopo; Relative Abundance 1



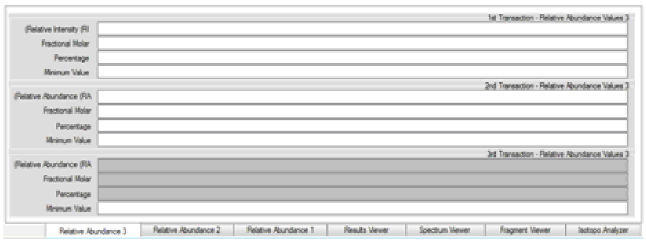
(c): Isotopo; Spectrum Viewer



(r): Isotopo; Relative Abundance 2



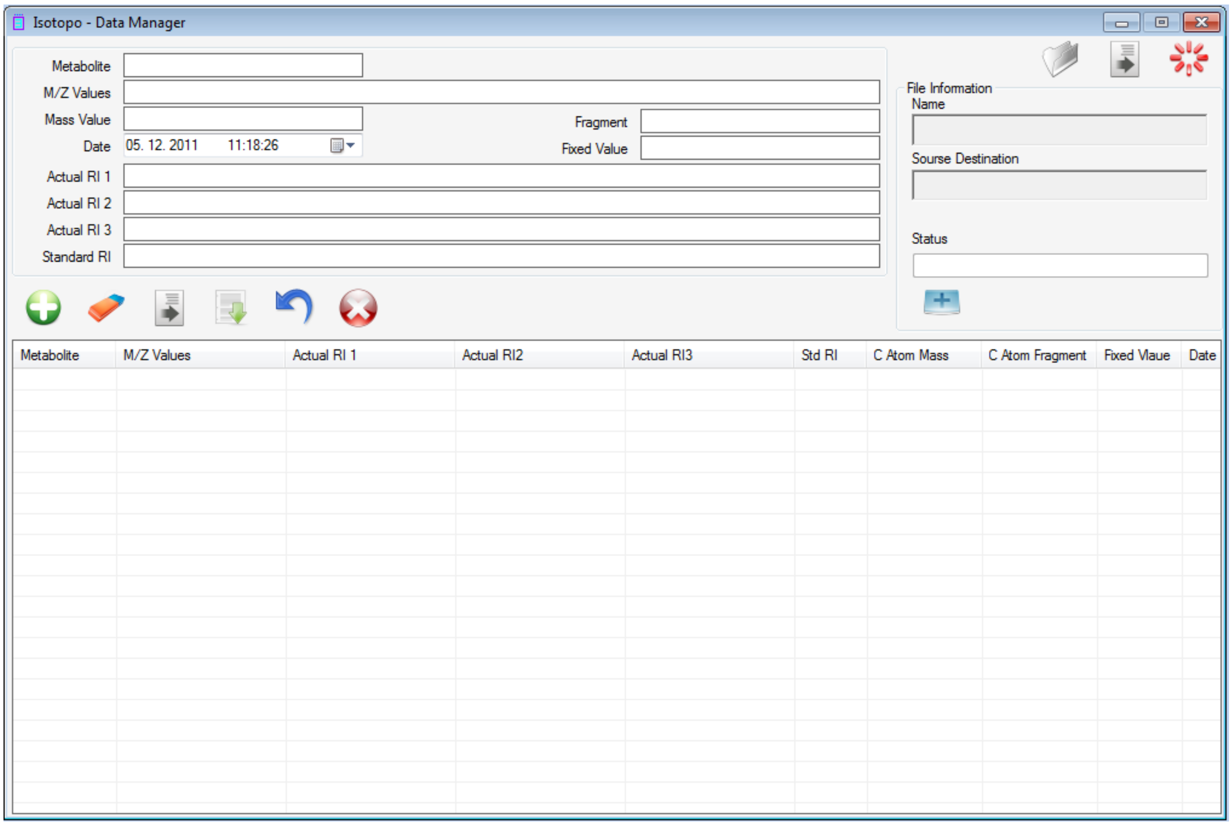
(d): Isotopo; Result Viewer



(g): Isotopo; Relative Abundance 3

Figure Legend. The Isotopo GUI of Data Analyzer (a) presents the main graphical user interface responsible for handling user data input, analyzing and producing spectrum, along with (b): Isotopo; Fragment Viewer, (c): Isotopo; Spectrum Viewer, (d): Isotopo; Result Viewer, (e): Isotopo; Relative Abundance 1, (r): Isotopo; Relative Abundance 2, (g): Isotopo; Relative Abundance 3.

Figure 10. Isotopo; Designed Main and Analyzer Graphical User Interface



Date: 05. 12. 2011 11:18:26

Metabolite	M/Z Values	Actual RI 1	Actual RI2	Actual RI3	Std RI	C Atom Mass	C Atom Fragment	Fixed Viaue	Date

Figure Legend. The Isotopo GUI of Data Manager presents the main graphical user interface responsible for handling user data input and providing data management and manipulation options.

Figure 11. Isotopo; Designed Data Manager Graphical User Interface

5. Conclusions

In this manuscript we have presented newly proposed and formally modelled bioinformatical software for mass isotopomers distribution analysis.

This manuscript thus provides guidance in advance for using the release of the architected meta-models and designed user interface of Isotopo

Now we are looking forward to successfully implement designs using V-Model software development model.

ACKNOWLEDGEMENTS

We (authors) are thankful to the Department of Bioinformatics, Biocenter, University of Wuerzburg Germany for giving us the opportunity to work on this research.

We are thankful to the editor for considering, blind reviewers for reviewing and publishers for publishing this manuscript.

We thank German Research Foundation (DFG)[TR34-Z1] for funding this research.

Author’s Contribution

Zeeshan Ahmed architected software designs and drafted manuscript as the First and Corresponding Author. Saman Majeed contributed in writing of the manuscript as co-author. Prof. Thomas Dandekar Lead and guided the study.

All authors participated in evaluation of the architecture designs as well as writing of the manuscript.

REFERENCES

[1] Zeeshan Ahmed, Saman Majeed, “Machine Learning and Data Optimization using BPNN and GA in DOC”, International Journal of Emerging Sciences, vol. 1, no. 2, pp.108-119, 2011.

[2] Saman Majeed, "Towards the Contribution of NEMDBs in Global Genetic Heterogeneity", International Journal of Emerging Sciences, vol.1, no.3, pp.433-443, 2011.

[3] Zeeshan Ahmed, Thomas Dandekar, Saman Majeed, "ADAM: Potential of PDM into Clinical Patient Data Management", International Journal of Emerging Sciences, vol.2, no.2, pp.280-299, 2012.

[4] Zeeshan Ahmed, Saman Majeed, “Middleware Technologies; Chain Web Grid Services”, International Journal of Web Applications, vol.3, no.4, pp.197-205, 2011.

[5] Zeeshan Ahmed, Thomas Dandekar, Saman Majeed, “Semantic web; Ontology Specific Languages for Web Application Development”, International Journal of Web

Applications, vol.4, no.1, pp.33-41, 2012.

[6] Hellerstein MK, Neese RA, “Mass isotopomer distribution analysis at eight years: theoretical, analytic, and experimental considerations”, Am. J. Physiol., vol.276, vol.6, pp.1146-1170, 1999.

[7] Scott E. Van Bramer, “Introduction to Mass Spectrometry”, Widener University, Department of Chemistry, September 2, 1998.

[8] Hites RA, “Gas chromatography mass spectrometry”, Handbook of instrumental techniques for analytical chemistry, ed. F.Settle. Upper Saddle River, N. J.: Prentice Hall. pp.609–626, 1997.

[9] Lee WN, Byerley LO, Bergner EA, Edmond J, “Mass isotopomer analysis: theoretical and practical considerations”, John Wiley & Sons, Inc., vol.20, no.8, pp.451-458, 1991.

[10] Korzekwa K, Howald WN, Trager WN, “The Use of Brauman’s Least Squares Approach for the Quantification of Deuterated Chlorophenols”, Biomed. Environ. Mass Spectrom., vol.19, pp.211-217, 1999.

[11] Schlatter R, Philippi N, Wangorsch G, Pick R, Sawodny O, Borner C, Timmer J, Ederer M, Dandekar T, “Integration of Boolean models exemplified on hepatocyte signal transduction”, Briefings in Bioinformatics, in press, 2011.

[12] Schwarz R, Musch P, von Kamp A, Engels B, Schirmer H, Schuster S, Dandekar T, “YANA - a software tool for analyzing flux modes, gene-expression and enzyme activities”, BMC Bioinformatics, vol.6, pp.135-146, 2005.

[13] Schwarz R, Liang C, Kaleta C, Kühnel M, Hoffmann E, Kuznetsov S, Hecker M, Griffiths G, Schuster S, Dandekar T, “Integrated network reconstruction, visualization and analysis using YANAsquare”, BMC Bioinformatics, vol.8, pp.1-10, 2007.

[14] Wolfgang Eisenreich, Jörg Slaghuis,Ralf Laupitz, Johanna Bussemer, Jochen Stritzker, Christine Schwarz, Roland Schwarz, Thomas Dandekar, Werner Goebel, Adelbert Bacher, “13C isotopologue perturbation studies of Listeria monocytogenes carbon metabolism and its modulation by the virulence regulator PrfA”, Proc. Natl. Acad. Sci. USA, vol.103 no.7, pp.2040-2045, 2006.

[15] Zeeshan Ahmed, Thomas Dandekar, Saman Majeed, “Unified Modeling and HCI Mockup Designing towards MIDA", International Journal of Emerging Sciences, vol.2, no.3, pp. 361-382, 2012.

[16] Brauman JI, “Least Squares Analysis and Simplification of Multi-Isotope Mass Spectra”, Anal. Chem., vol.38, no.4, pp.607–610, 1996.

[17] Coolidge JL, “The Story of the Binomial Theorem”, The American Mathematical Monthly, vol.56, no.3, pp.147–157, 1949.

[18] Coolidge JL, “The Story of the Binomial Theorem”, The American Mathematical Monthly, vol.56, no.3, pp.147–157, 1949.

[19] Alston S. Householder, “The theory of matrices in numerical analysis”, Dover Publications, 1975.

[20] Roger A. Horn, Charles R. Johnson, “Matrix Analysis”, Cambridge University Press,1990.

[21] Brillinger DR, “The identification of a particular nonlinear time series system”, Biometrika, vol. 64, no.3, pp.509-515, 1977.

[22] Zeeshan Ahmed, “Towards Performance Measurement and Metrics based Analysis of PLA Applications”, International Journal of Software Engineering & Applications, vol.1, no.3, pp.66-80, 2010.

[23] Medvidovic N, Rosenblum DS, Redmiles DF, Robbins JE, “Modeling software architectures in the Unified Modeling Language”, ACM Trans. Software Engineering Methodologies, vol.11, no.1, pp.2-57, 2002.

[24] Ambler SW, “The Elements of UML 2.0 Style”, Cambridge University Press, 2005.

[25] Paul Rook, “Controlling software projects”, Software Engineering Journal - Controlling software projects, vo.1, no.1, pp.7-16, 1986.

[26] Mathur S, Malik S, “Advancements in the V-Model”, International Journal of Computer Applications, vol.1, no.12, pp.0975 – 8887, 2010.

[27] Jacobson I, Christerson M, Jonsson P, Övergaard G, “Object-Oriented Software Engineering: A Use Case Driven Approach”, Reading, MA: Addison-Wesley, 1992.

[28] Bruza PD, van der Weide T, “The Semantics of Data Flow Diagrams”, in Proceedings of 1993 the International Conference on Management of Data, 1993.

[29] Latronico E, Koopman P, “Representing Embedded System Sequence Diagrams as a Formal Language”, in Proceedings of 2001 4th International Conference on The Unified Modeling Language, Modeling Languages, Concepts, and Tools, 2001.

[30] Marilyn B, “A guide for programmers”, Prentice-Hall. 1978.

[31] Berardi D, Calvanese D, Giacomo GE, “Reasoning on UML class diagrams”, Artif. Intell., vol.168, no.1, pp.70-118, 2005.

[32] Grant ES, Chennamaneni R, Reza H, “Towards analyzing UML class diagram models to object-relational database systems transformations”. in Proceedings of 2006 24th IASTED international conference on Database and applications, 2006.

[33] Zeeshan Ahmed, Suhdir K. Ganti, Hans Kyhlbäck, "Design Artifact’s, Design Principles, Problems, Goals and Importance", in Proceedings of 2008 4th International Statistical Conference 2008.

[34] Klemmer SR, Lee B, “Notebooks that Share and Walls that Remember: Electronic Capture of Design Education Artifacts”, in Proceedings of 2005 ACM Symposium on User Interface Software and Technology, 2005.