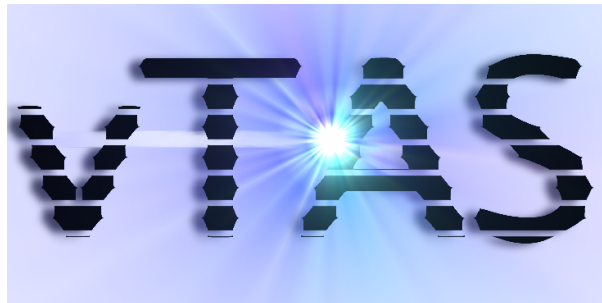




INSTITUT LAUE-LANGEVIN

SUMMER STAGE



Author:
Peter BRADEN

Supervisors:
Martin BOEHM
Alain FILHOL

August 31, 2007

Abstract

The triple-axis spectrometer is an instrument that uses neutron scattering to examine structural and magnetic excitation. It is not easy to use, however, as measurements must constantly be translated to, and from, reciprocal space - a mathematical space used to simplify calculation.

vTAS is a graphical program that aids this process by modeling the instrument, allowing easy experimentation with the configuration and sample parameters. It also maintains a representation of the instrument's physical state, for example the location of the walls around the instrument, and its angular limits.

This report aims to explain the processes behind the design and implementation of vTAS, some of the physics encountered when developing the program, and a summary of the author's work at the Institut Laue-Langevin over the summer stage.¹

¹Stage: French internship

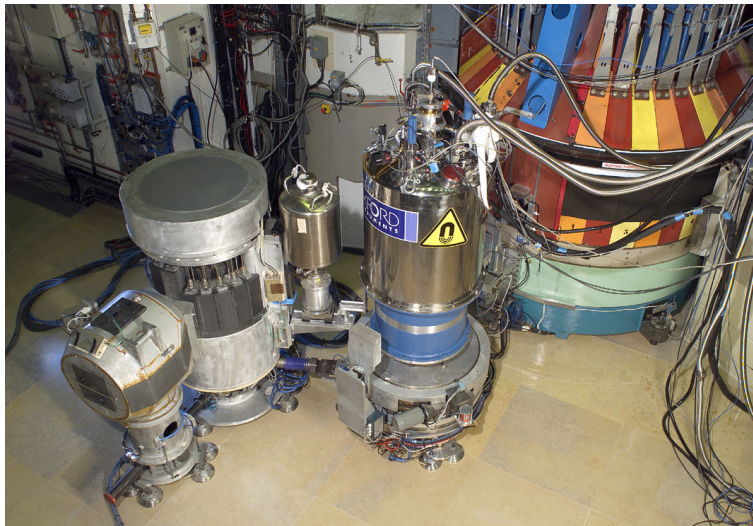


Figure 1: IN14 - One of the triple-axis spectrometers at the ILL
From left to right: detector (see 2.4.5), analyser (see 2.4.4), spin flipper (flip the neutrons spin state), cryomagnet containing the sample (cools and applies a magnetic field to the sample), monochromator housing (see 2.4.2). The reactor is behind the monochromator housing.

Contents

1	Introduction	5
2	The Physics of the Triple Axis Spectrometer	6
2.1	Introduction	6
2.2	Neutron Scattering	6
2.3	Solid State Physics	7
2.3.1	Crystal Structure	7
2.3.2	Reciprocal Space	9
2.4	Components of the Triple Axis Spectrometer	9
2.4.1	The Source	9
2.4.2	The Monochromator	9
2.4.3	The Sample	10
2.4.4	The Analyser	10
2.4.5	The Detector	11
3	The Software Engineering behind vTAS	12
3.1	Introduction	12
3.2	Existing Program	12
3.3	Specification	13
3.4	New Features	13
3.4.1	Print	13
3.4.2	Load/Save	13
3.5	Design	14
3.5.1	Architecture	14
3.5.2	User Interface	15
3.5.3	Mathematical Model	17
3.6	Internationalisation	19
3.7	Documentation of the Project	20
3.8	Testing	20
4	Evaluation	21
4.1	Introduction	21
4.2	Critical Evaluation of the Program	21
4.2.1	Problems Encountered	22

4.3	Remaining Tasks	22
4.4	Areas for Extension	22
Appendices		24
A Definition of Variables		25
A.1	Angles	25
A.2	Experiment	25
A.3	Sample	26
A.4	Other Parameters	26
A.5	Coordinates of Instrument Components	26
B Formulae		28
B.1	Calculate Sample Matrices from Sample Parameters	28
B.1.1	Unit Cell Volume	28
B.1.2	Reciprocals	28
B.1.3	UB Matrix	29
B.2	Calculate Angles from Experiment Values	29
B.2.1	Translate \vec{Q} into reciprocal space	29
B.2.2	Use inverse Bragg law to calculate a_1 & a_2	29
B.2.3	Use cosine rule to calculate a_4	29
B.2.4	Calculate a_3 using vector triangle	30
B.2.5	Use inverse Bragg law to calculate a_5 & a_6	30
B.3	Calculate Experiment Values From Angles	30
B.3.1	Wavelengths	30
B.3.2	Calculating \vec{Q}	30
B.3.3	Calculating \vec{K}_i	30
B.3.4	Calculating ΔE	31
B.4	Plot Triple Axis Spectrometer from Angles and Lengths	31
B.5	Calculate Angles from the Positions of Spectrometer	31
C Submitted Files		32
C.1	Code	32
C.2	Documentation	32
C.3	Web Page	32

List of Figures

1	IN14 - One of the triple-axis spectrometers at the ILL	1
2.1	A comparison of the radiographs of neutrons(left) and x-rays(right).	6
2.2	The crystal unit cell, showing the relationships between the angles and vectors.	7
2.3	The seven types of crystal system	8
2.4	The positions of vectors and angles within reciprocal space. . . .	10
2.5	The location of angles within the system.	11
3.1	The structure of the data model.	14
3.2	The dependencies of the components within the Model-View- Controller structure.	16
3.3	A screenshot of the main window of vTAS	17
3.4	A sample of the document produced by the print function of the program.	18
3.5	The relationships and dependencies between the variables when calculating angles from reciprocal triangle.	19
4.1	Code Metrics	22
A.1	The relationships and dependencies that need to be considered when calculating experiment values from angles.	27

Chapter 1

Introduction

The Institut Laue-Langevin (ILL) in Grenoble is home to the most intense neutron source in the world. The neutrons it produces are used in the study of a wide range of subjects, from biology and particle physics, to material science. One class of instruments used in these experiments is the triple-axis spectrometer, an instrument that allows for the precise measurement of energy and momentum transfer from the neutron to the sample, or from the sample to the neutron, during neutron bombardment.

The mathematics to translate the orientations and energies of an experiment into parameters suitable for the instrument are tedious, therefore to aid in the design of experiments, a computer program is necessary.

The first version of vTAS was written in 1998 by Alain Bouvet, a physicist at the ILL at the time. The program was written in java with an `awt` interface. Unfortunately the program was very buggy, and its cramped interface made it almost impossible to use.

In the summer of 2006 the program was resurrected by Noelle Le Delliou and given a new interface using the `swing` toolkit. The program still contained a number of errors, and by now the code had fallen into such disarray that further progress was unfeasible.

It was therefore decided to completely reimplement the program based on modern software engineering principles, allowing for the inclusion of many new features.

Chapter 2

The Physics of the Triple Axis Spectrometer

2.1 Introduction

Clearly, when designing software for an instrument such as the triple axis spectrometer, some grasp of the underlying physics of the instrument must be attained. This chapter aims to look at the physics behind the triple axis spectrometer, and how it relates to the calculations that are performed in modeling the instrument.

2.2 Neutron Scattering

Neutron Scattering is a method of analysing the structure and excitation of a material by observing the diffraction of a beam of neutrons. As neutrons have no charge, they can be used to measure the structure of charged substances because the neutron is able to penetrate the ‘wall’ of charge surrounding atoms. As neutrons also have spin, they can be used to measure samples with subtle magnetic fields at an atomic scale.

By exploiting the properties of neutrons, a very detailed picture of the structure of a sample, and the stresses and strains within it, can be built up in a non-intrusive manner. Furthermore, as the neutrons penetrate the sample, they can be used to probe the structure inside a sample.

Neutron probing techniques are comparable to X-ray analysis, as is performed at synchrotron facilities such as the ESRF¹. The



Figure 2.1: A comparison of the radiographs of neutrons(left) and x-rays(right).

¹European Synchrotron Radiation Facility - An advanced synchrotron X-ray source next

techniques are complementary as neutrons are very good at analysing the positions of weakly charged particles such as hydrogen atoms, whereas X-rays work better with strongly charged particles. Figure 2.1 shows an example of a comparison of the two techniques. Note that the neutrons provide a better picture of the plastic parts of the camera - those that contain hydrogen, whereas the x-rays are better at imaging the metals.

2.3 Solid State Physics

Although neutron scattering can be used to analyse a wide range of samples, the instruments which vTAS is designed to model are concerned mainly with the study of solids, in particular solids with a crystalline substructure.

2.3.1 Crystal Structure

Crystals have, by definition, a periodic structure. This structure greatly simplifies the mathematics needed to describe the sample, and therefore facilitates the study of the material. A consequence of this is that crystals are the most researched type of samples within solid state physics, and thus the study of crystal structure is a mature field.

The periodicity of crystals means that a large crystal structure can be simplified down into a single repeating unit. This *unit cell* allows the positions of all of the atoms in the sample to be specified in a relative manner. This topic is more complex than can be described here, however it is covered in great detail in [5]. For the purpose of this report we will simply define the unit cell of a crystal to be a set of three vectors A , B and C at angles α , β and γ , that specify the edges of the crystal's periodic unit (see fig 2.2).

There is only a finite set of crystals that are possible, and by characterising groups of these variations by their orders of symmetry, we can classify all crystals into seven *crystal systems* (fig 2.3).

In addition to these parameters, when studying a crystal with a triple axis spectrometer we must also specify the orientation of the lattice in respect to the instrument axes. As this technique analyses a two dimensional scattering plane, only a cross section of the crystal can be analysed. This plane is denoted by two vectors \vec{O}_1 and \vec{O}_2 .

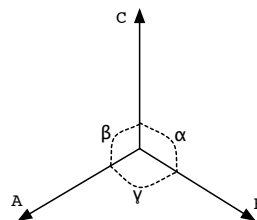


Figure 2.2: The crystal unit cell, showing the relationships between the angles and vectors.

to the ILL

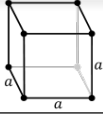
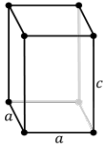
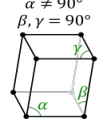
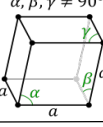
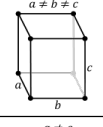
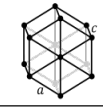
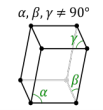
	Cubic
	Tetragonal
	Monoclinic
	Rhombohedral
	Orthorhombic
	Hexagonal
	Triclinic

Figure 2.3: The seven types of crystal system

2.3.2 Reciprocal Space

One of the mathematical devices utilised in simplifying the calculations from the data gained from the experimentation is to convert the sample lattice into reciprocal space and relative units. Reciprocal space is a space in which dimensions are the reciprocals of those in real space. Relative units are used to scale the reciprocal space to allow measurements in terms of the unit cell of the crystal, rather than absolute dimensions. These conversions allow us to simplify the formulae that describe phenomena related to the scattering. To convert between absolute units and relative units we use a matrix, known as a *UB matrix*, which is calculated from the unit cell parameters and the two vectors specifying the samples orientation[Eqn. B.1.3].

We can use this reciprocal lattice to plot the crystal's interactions with the neutron beam. By defining vectors \vec{K}_i and \vec{K}_f as vectors with the direction of the incoming and outgoing beams to the sample respectively, and lengths equal to the reciprocal of the corresponding beam's wavelengths, we can model the scattering of the beam as a vector triangle[fig. 2.4]. The third side of the triangle, which we define as \vec{Q} , is therefore the vector difference caused by the scattering. This vector is equivalent to the momentum transfer and is used in further calculations.

2.4 Components of the Triple Axis Spectrometer

The triple axis spectrometer is comprised of several components, mechanically movable in relation to each other along three axes.(fig. 2.5)

2.4.1 The Source

To probe materials accurately, a reliable source of neutrons must be used. In the ILL, this source takes the form of a nuclear reactor. The neutrons emitted from the nuclear fission of the fuel rods are channeled to the experiment along guides, similar to optical fibres.

Other facilities produce neutrons by other means, including spallation which produces a short burst of neutrons by colliding photons with heavy atoms (ie. lead, tungsten).

The exact workings of the source are unimportant to the triple axis spectrometer, and all that needs to be noted is that the source produces neutrons with different energies.

2.4.2 The Monochromator

The closest component to the neutron source is the monochromator. The function of this component is to extract a neutron beam with a well defined wavelength. This is achieved by exploiting Bragg diffraction - the diffraction of the beam by a crystal lattice with regularly spaced lattice layers D_m apart. The

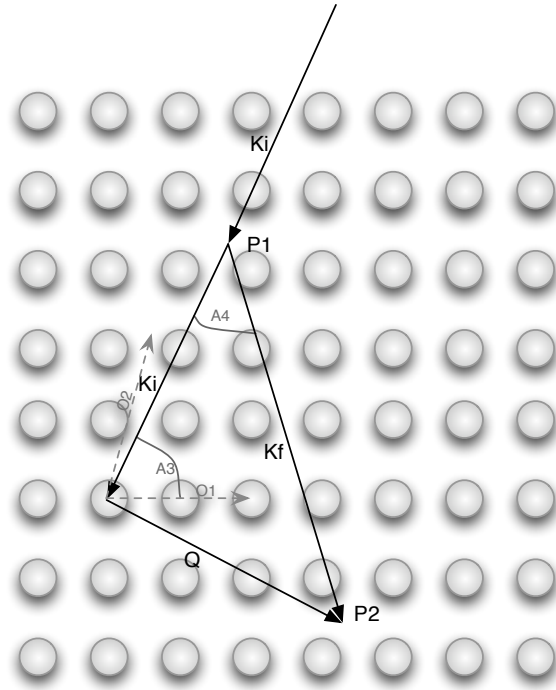


Figure 2.4: The positions of vectors and angles within reciprocal space.

wavelength emitted from the monochromator can be calculated with the Bragg Law:

$$n\lambda = 2D_m \sin \theta \quad (2.1)$$

In this case we can ignore harmonics, meaning that n can be assumed to be 1.

2.4.3 The Sample

As has been mentioned, although many types of material can be analysed by the triple axis spectrometer, we are interested in solids with a crystal substructure. We will therefore assume that the sample within vTAS is a crystal with a specified unit cell, oriented within the system to a plane specified by the two vectors O_1 and O_2 .

2.4.4 The Analyser

The analyser is similar to the monochromator. It is used to analyse a specific energy transfer within the scattered neutrons.

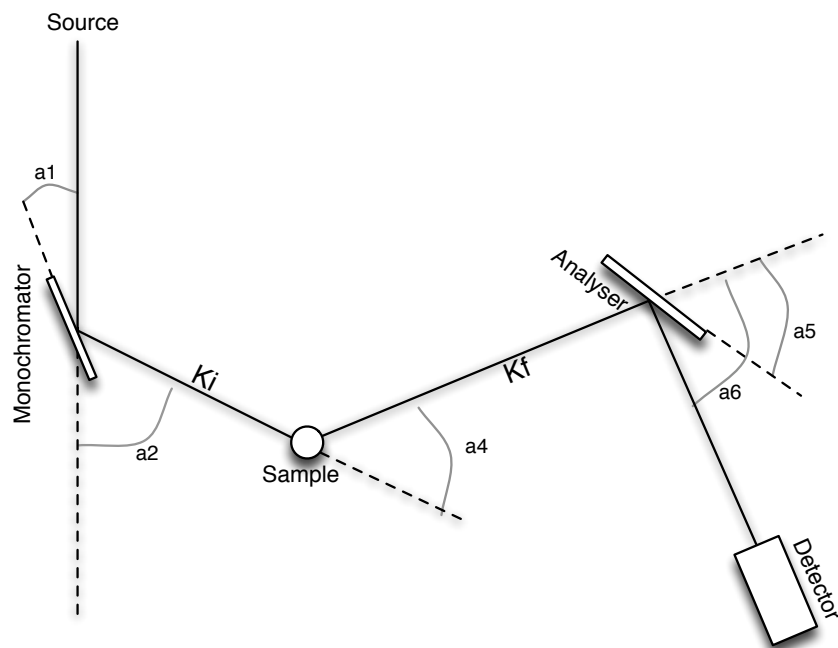


Figure 2.5: The location of angles within the system.

2.4.5 The Detector

The detector simply measures the intensity of the beam output by the analyser. Typically the detector scans with the analyser through a range of angles measuring intensity at each. As this is a fairly slow process, even with a source as intense as that at the ILL, several detectors may be combined, for example in the *Flat Cone* detector which combines 31 detectors into a single unit allowing for much quicker measurement.

Chapter 3

The Software Engineering behind vTAS

3.1 Introduction

As has been discussed in the previous chapter, the physics behind vTAS are not straightforward. The design of software to help model such a scenario must therefore be built upon solid software design principles to allow as much effort as possible to be channeled into understanding the complexities of the mathematics behind the design.

vTAS is a fairly complex entity, comprising thousands of lines of code in a structured and layered architecture. Combined with the fact that the author had no experience in solid state physics before working on the project, and the vagueness of the specification, the task was an ambitious one. This section aims to look at the creation of the program from a software engineering perspective.

3.2 Existing Program

The original version of vTAS was written by Alain Bouvet in 1998. Developed in an early version of `awt`, it contained many bugs, and was almost impossible to use. Many of the formulae were literal translations from fortran. The code was contained in a single monolithic class, over 3200 lines long, and, as java was a new language and its object oriented nature was not as widely known, was written in a functional style. The program did, however, contain many of the features still in the modern version. Both the instrument preview pane and the reciprocal space pane were based on visualisations offered in this initial version.

In 2006 it was decided to update the program. Noelle De Delliou was tasked with modernising the interface and improving the ease of use of the program. The existing codebase was utilised, however, which meant that the project was hampered by the disorganisation of the code.

3.3 Specification

The specification for the third version of vTAS was loosely defined, therefore, as follows:

Rewrite the program, to follow modern software engineering principles, thus allowing for the future addition of features, and improving the user experience.

The vagueness of the specification meant that a flexible approach to the development of the program was necessary. In addition, as the specification became clearer as the project developed, an agile development paradigm was necessary in order to adapt to the current requirements. As the vTAS was only being developed by a single person, however, a rigid software engineering model was not of great importance to the project, and thus the project was undertaken with only a basic development model, based partly on the author's experience with the extreme programming paradigm.

3.4 New Features

The program itself has been completely rewritten. Although the majority of the time was spent replicating the underlying data model, and the visualisations from the previous program, a few new features were added.

3.4.1 Print

As it may be necessary to calculate parameters with vTAS and then use them in other locations, a print function was deemed a worthwhile addition to the program. The print function is fairly basic, merely printing the state of the spectrometer along with the visualisations, however it summarises all of the programs data, and the code for the feature has the possibility to be extended should additional functionality be required.

3.4.2 Load/Save

Rather than have to set up the program every time it is run, load/save functionality means that the state of the program can be persistent between executions. Save files could potentially be emailed between users of vTAS, allowing an element of collaboration.

The file format for vTAS save files was deliberated over. XML¹ was a logical choice as it has become the de-facto standard for data storage. Instead of creating a new schema for the format, it was decided to utilise the java `Properties` class which allows for the simple export and import of data to an xml file. This

¹XML: eXtensible Markup Language - a file format that allows for the semantic storage of character data

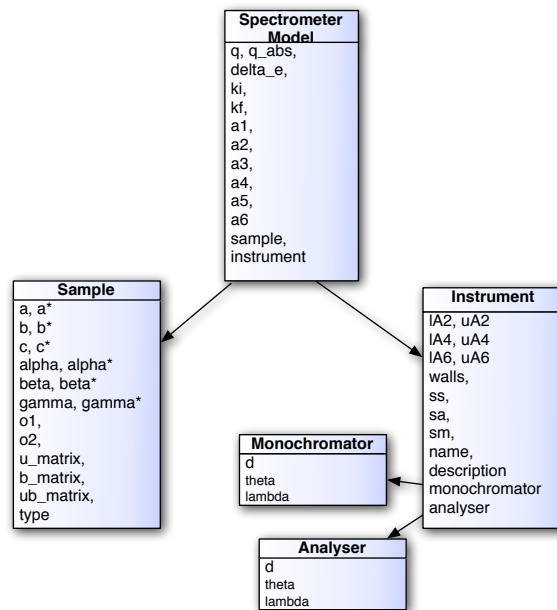


Figure 3.1: The structure of the data model.

means that the actual format of the file can be left to the java implementation, a level of abstraction that greatly eases the task of managing the program's files.

The load and save feature is rather temperamental at present, due to insufficient time to finish and test it, however it should not be a hard task to extend it as most of the functionality is in place.

3.5 Design

One of the key flaws with the previous program was that the code had been assembled disparately throughout a long period of time and without an idea of the final form of the program. This meant that the code was convoluted and, in some places, almost incomprehensible. One of the key aims of the new version of vTAS, therefore, was to improve the design of the program.

3.5.1 Architecture

In complex applications where information is presented to the user, and can be manipulated, it is considered good practice to separate data from the user interface. This provides several benefits. Firstly, it allows for increased modularity of code. As the application is split into several discrete layers, each layer can be

thought of as a distinct component. This encapsulation, one of the key tenets of object oriented design, improves the flexibility and durability of the program, as each component can be tested and developed separately.

Separating the view from the data model can also allow the same data to be displayed in several different ways. In vTAS, the same underlying data is used to display several visualisations, and it is this separation of data and presentation that allows this to be so flexible, extensible and organised.

One of the popular methods of separating data from presentation is known as Model-View-Controller. In this architecture, the data is represented within a *model*, the user interface is known as the *view*, and an intermediary layer, known as the *controller*, is introduced to facilitate communication between the two. Figure 3.2 shows how this architecture is applied to vTAS. As the layers are complex, in vTAS they are organised into hierarchies.

This architecture lends itself to several design patterns. As the controller layer is constantly monitoring the other layers for change, the Observer/Observable design pattern is an ideal pattern to use. As the Swing toolkit already allows for this pattern, by implementing action listeners, only the observation of the data model has to be explicitly defined.

Another design pattern that was utilised within the program was the singleton pattern. The singleton pattern is a design pattern that aims to simplify the referencing of an object within a program when it will only ever exist once. This is the case for all three of the MVC layers in vTAS, and therefore the root node of each layer's hierarchy was implemented as a singleton. This means that the reference for any object within the program can be obtained simply by locating the root node and then traversing the layers tree. This technique allows great flexibility, and ease of design however it should be used judiciously as without care it can lead to complicated code.

3.5.2 User Interface

The program's specification dictated that the swing toolkit should be used for the graphical user interface of the program. Swing is a lightweight, platform independent interface toolkit for java. This requirement left great scope for the program. The previous versions of the program had used an applet architecture, however this was not regarded as a benefit as it left the program constrained by the performance and space constraints of the browser. By promoting the program to be a full desktop application, more freedom was allowed in designing the interface. To fulfil the need to provide a runnable version of the program on the internet, **Java Web Start** was used to allow the program to be run from a hyperlink.

With the constraints of the applet gone, the design of the application was ready to be overhauled. One of the criticisms of the previous interface was that the reciprocal space panel was too small to be used easily. In the new design this has been enlarged significantly, and by changing the magnification with a zoom bar or the scroll wheel of the mouse the panel has a much simpler interface.

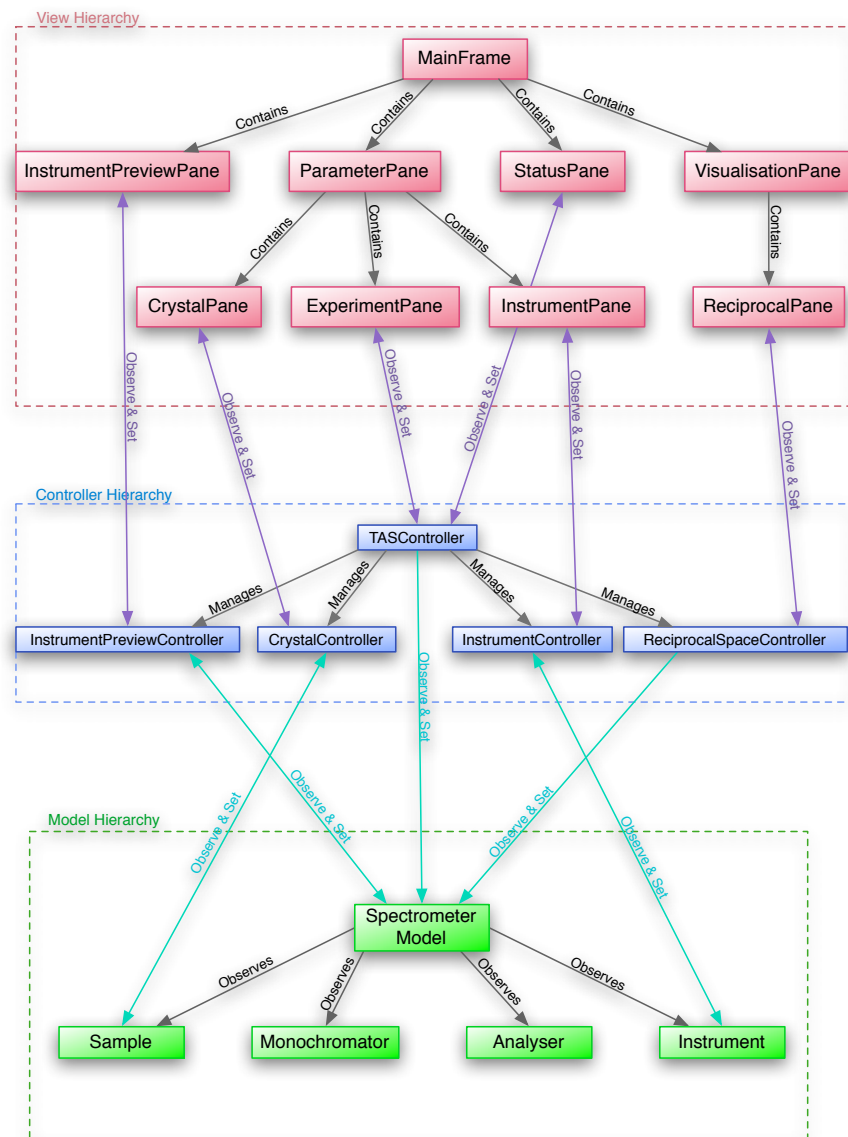


Figure 3.2: The dependencies of the components within the Model-View-Controller structure.

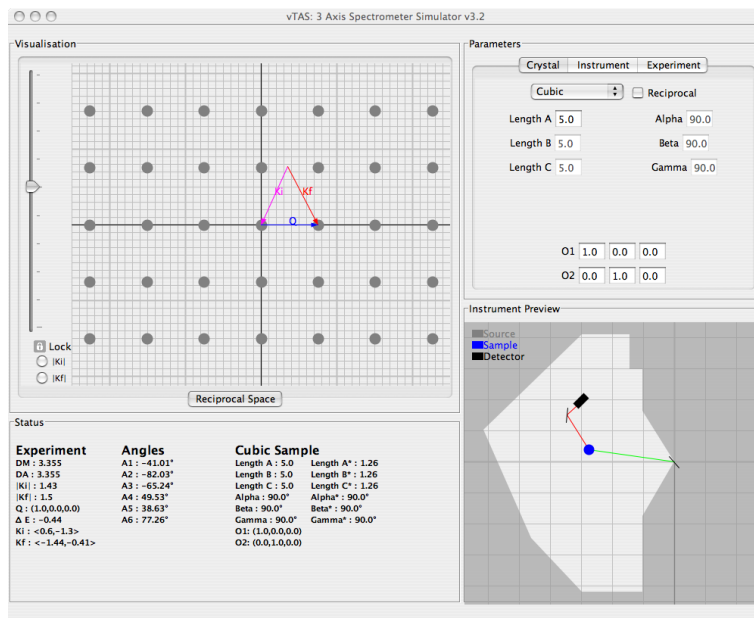


Figure 3.3: A screenshot of the main window of vTAS

The previous program placed a great deal of importance on the preview of the instrument, giving it almost half of the window’s space. This was very wasteful, and therefore in the new program this has been shrunk down, with the instrument automatically scaled to fit.

Another criticism of the previous design was that there was no indication of what could be edited or moved within the visualisations. In the current version of the program this is addressed by having small movement arrows appear over movable parts when the mouse enters the appropriate pane.

3.5.3 Mathematical Model

One of the flaws of the previous incarnations of the program was that the mathematics was implemented without taking advantage of the object oriented design of the language. This led to very convoluted code, as formulae were frequently reused.

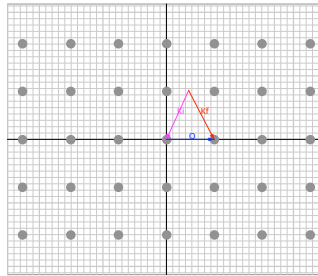
In response to these difficulties, one of the first things to be written was a set of data type classes for commonly used types, for example angles and vectors. These data type classes are immutable, which means that they can be passed by reference, like strings, without the integrity of the data being compromised.

The actual data model, like the other components of the program, is organised in a hierarchy. At the top of the tree is the `Spectrometer Model` class. This class implements the interface `ISpectrometerModel` which specifies the methods by which the model can be altered. It also provides references to the

vTAS Configuration

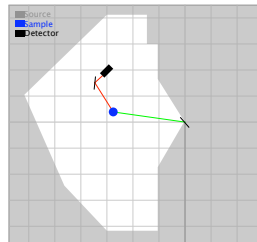
Experiment

|K_{il}: 1.43
 |K_{fi}: 1.5
 Δ E: -0.44
 Q: (1.0,0,0,0)



Cubic Sample

Length A: 5.0	Length A': 1.26
Length B: 5.0	Length B': 1.26
Length C: 5.0	Length C': 1.26
Alpha: 90.0°	Alpha': 90.0°
Beta: 90.0°	Beta': 90.0°
Gamma: 90.0°	Gamma': 90.0°



Angles

A1: -41.01°
 A2: -82.03°
 A3: -65.24°
 A4: 49.53°
 A5: 38.63°
 A6: 77.26°

Figure 3.4: A sample of the document produced by the print function of the program.

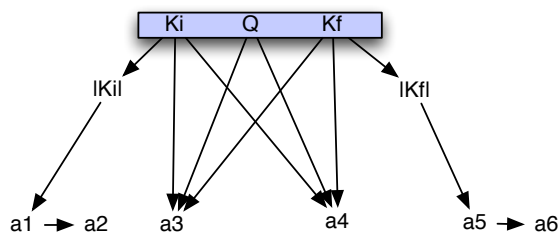


Figure 3.5: The relationships and dependencies between the variables when calculating angles from reciprocal triangle.

various components of the system, namely the monochromator, analyser, sample and instrument.

Most of the mathematics in vTAS is related to trigonometry. The visualisations of the underlying data model rely heavily on the elementary properties of triangles in their calculation, and the calculations are largely derived from the trigonometry of the vector triangle within reciprocal space. Many of the more complex of the formulae used in the model can, in fact, be derived from the cosine law. It may have therefore been simpler, when programming the data model, to have tackled the mathematics in a more semantic way. Currently the formulae are implemented as lines of mathematical code, generally using primitive data types. If the data model was to be rewritten it may be advantageous to consider the problem as a trigonometric one, and thus infer the benefits of object orientation and strong typing within the calculations, rather than simply translate mathematical formulae to code as exists in the current system.

An attempt was made to convert the model in this way however time constraints prevented this from being successfully completed. This would be a good area for extension as by converting to an object based model, increased reliability in the mathematics, and modularity of variables, could be accomplished.

3.6 Internationalisation

In an international facility such as the ILL, many languages are spoken. Accordingly, when designing a program to be used in such an environment, internationalisation (often abbreviated as i18l) must be given due consideration. The previous program contained many hard coded constants scattered throughout the program. This is considered bad programming practice as it makes the program difficult to translate and requires the program to be recompiled for each new language.

Java provides tools to facilitate the internationalisation of programs in the form of the `ResourceBundle` class. By storing the program's variables within an external resource file and then loading them at runtime, the program can be easily run in several languages.

The current version of vTAS utilises these resource bundles, allowing the program to be run in English or French, depending on the operating system's default language. Furthermore, as the constants exist within a UTF8² encoded text file, new languages can be added without recompiling.³

3.7 Documentation of the Project

One of the negative aspects of the previous code-base was the lack of meaningful documentation accompanying it. At several points throughout the code, important formulae were interspersed with user interface code with no commenting indicating their function.

Documentation is a crucial part of the modern software engineering discipline. As software projects are likely to be developed by several people over their lifespan, some explanation of their functionality is vital to provide subsequent developers with the knowledge necessary to add to the code.

In the development of vTAS it was necessary, therefore, to provide a high level of technical documentation. As the program was being developed in java, it was logical to use the standard java documentation format, `javadoc`, to create the documentation.⁴

The documentation, along with this report, should allow future developers to gain some insight into the way the program works, and locate any errors that may occur in the mathematics.

3.8 Testing

To properly verify the mathematics of the program, a suite of tests was written to compare the output of the program with `TasMAD`⁵, and therefore ensure their veracity. The tests are not as extensive as they could be, however they provide a good level of confidence in the correctness of the data model layer.

²UTF8 is a character encoding, similar to ASCII, that allows the inclusion of any character in the unicode standard which contains most international characters.

³ For more information on this method of internationalisation, the reader is encouraged to consult the official java documentation[8] , or the java tutorial[9].

⁴Javadoc is an automatic documentation generator that processes comments within the source files.

⁵TasMAD: The instrument control software

Chapter 4

Evaluation

4.1 Introduction

In a project as ambitious as this one, it is inevitable that some tasks will remain unfinished, and in some areas the code will not be as polished as that which was initially envisioned. This section aims to take an objective look at the project, discuss some of the areas which led to difficulties, and give a balanced overview of the project.

4.2 Critical Evaluation of the Program

In a summer vTAS has been completely reimplemented. Although some bugs still remain in the code, the submitted program is functional and hopefully will prove useful to the users of the triple axis spectrometer. Moreover, the program is now written to a high standard, combined with an architectural plan. These qualities make it trivial to add functionality or additional visualisations. In these respects vTAS is a great success. Although not much completely new functionality was added to the program, many small changes were made, including visual indications of the angular limits, and improvement to the zoom function within the reciprocal pane. In addition the codebase is now a solid foundation on which future programs can be built.

One of the features that was requested to be added over the summer was a simulator for the flat cone detector. Unfortunately, the limited amount of time in the summer meant that this was too ambitious an objective. Another feature which was requested but not completed was an E vs. Q visualisation. These features should be easily implemented, and could provide a specification for a future summer stage.

Lines of Code	5873
Number of Classes	65
Number of Methods	535
Number of Overridden Methods	50
Mean Afferent Coupling	6.037
Mean Efferent Coupling	1.926

Figure 4.1: Code Metrics

4.2.1 Problems Encountered

One of the most persistent problems faced during the development of vTAS was the complexity imposed by the large number of coordinate systems needed. The translation between the Cartesian plane and the coordinate system used within `swing`, although theoretically simple, made calculating the trigonometry required by the visualisations difficult. This problem was mitigated by reorganising the code in a way that made obvious the system in which a point resided. As mouse positions are reported within the `swing` coordinate system, at several points a two way translation has to be performed for a single calculation. It was thought that the efficiency of the program was less important, in this situation, than the clarity of the mathematics.

Another conceptual problem was the understanding of the difference between reciprocal space and the difference between relative and absolute units. This confusion, and an uncertainty as to which variable lay in which space and in which units meant that a lot of the formulae had to be rewritten.

4.3 Remaining Tasks

The program has a number of areas that are incomplete. These should obviously be treated with the greatest importance when future work is to be done on vTAS. The preferences pane, although mostly functional, still needs work, for example on the table of wall coordinates. The angles within the preview pane are the negative of the angles within the system. Modifying these may mean that the formulae to plot the points within the pane may need to be updated.

The load/save functionality is still a little temperamental. It is able to retrieve state for the experiment however instrument parameters, for example the walls, are not retrieved from the save file. In addition, errors are sometimes encountered when loading the save files.

4.4 Areas for Extension

The program has many opportunities for extension. The requested flat-cone and E vs. Q visualisations should be treated as of high importance for a future stage. Additionally finishing the load/save functionality would also be desirable. Based

on the author's experience over the summer, there is at least another summer of work that could be completed on the program. A suitable specification for such a stage could be as follows:

Extend vTAS to include additional features and functionality such as flat cone detection and an E vs. Q visualisation, and additionally to complete existing features such as load/save.

Other features that may be desirable in future versions of vTAS could include: undo functionality, where the user is able to undo an erroneous modification; better error reporting, should the angles be out of accepted bounds or should \bar{Q} not be within the reciprocal plane; the implementation of more instruments and facilitating the modification of the existing ones.

Appendices

Appendix A

Definition of Variables

A.1 Angles

- a_1 Angle between monochromator and incoming beam.
- a_2 Angle between incoming and outgoing beam in respect to monochromator.
- a_3 The angle between the incoming wave vector and the sample orientation vector \vec{O}_1 .
- a_4 Angle between incoming and outgoing beam in respect to the sample.
- a_5 Angle between analyser and incoming beam.
- a_6 Angle between incoming and outgoing beam in respect to the analyser.

A.2 Experiment

\vec{K}_i Incoming wave vector within reciprocal space $\rightarrow |\vec{K}_i| = \frac{2\pi}{\lambda_{incoming}}$ (Reciprocal Space, Absolute Units : \AA^{-1})

\vec{K}_f Outgoing wave vector within reciprocal space $\rightarrow |\vec{K}_f| = \frac{2\pi}{\lambda_{outgoing}}$ (Reciprocal Space, Absolute Units : \AA^{-1})

\vec{Q}_{abs} Difference vector within reciprocal space (Reciprocal Space, Absolute Units : \AA^{-1})

\vec{Q} Difference vector within reciprocal space (Reciprocal Space, Relative Units)

ΔE Energy change from collision (meV)

A.3 Sample

A Crystal unit cell dimension(Real Space, Absolute units: Å).

B Crystal unit cell dimension(Real Space, Absolute units: Å).

C Crystal unit cell dimension(Real Space, Absolute units: Å).

α Angle between B and C.

β Angle between A and C.

γ Angle between A and B.

\vec{O}_1 Vector indicating the samples orientation plane(Reciprocal Space, Relative Units).

\vec{O}_2 Vector indicating the samples orientation plane(Reciprocal Space, Relative Units).

A.4 Other Parameters

D_m Lattice Spacing of the monochromator.

D_a Lattice Spacing of the analyser.

S_m Scattering Sense of the monochromator.

S_s Scattering Sense of the sample.

S_a Scattering Sense of the analyser.

A.5 Coordinates of Instrument Components

M Monochromator

R Neutron Source

S Sample

A Analyser

D Detector

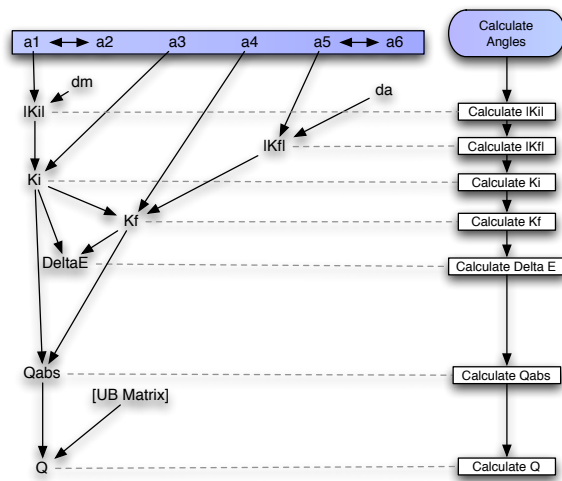


Figure A.1: The relationships and dependencies that need to be considered when calculating experiment values from angles.

Appendix B

Formulae

B.1 Calculate Sample Matrices from Sample Parameters

B.1.1 Unit Cell Volume

As all crystal systems are special cases of the triclinic system, we can calculate the volume using this single formula. [1]

$$Volume = A \cdot B \cdot C \cdot \sqrt{(1 - \cos^2 \alpha - \cos^2 \beta - \cos^2 \gamma + 2 \cos \alpha \cos \beta \cos \gamma)} \quad (\text{B.1})$$

B.1.2 Reciprocals

$$A^* = \left(\frac{B \times C \times \sin \alpha}{Volume} \right) \cdot 2\pi \quad (\text{B.2})$$

$$B^* = \left(\frac{A \times C \times \sin \beta}{Volume} \right) \cdot 2\pi \quad (\text{B.3})$$

$$C^* = \left(\frac{A \times B \times \sin \gamma}{Volume} \right) \cdot 2\pi \quad (\text{B.4})$$

$$\alpha^* = \arccos \left(\frac{\cos \beta \cos \gamma - \cos \alpha}{\sin \beta \sin \gamma} \right) \quad (\text{B.5})$$

$$\beta^* = \arccos \left(\frac{\cos \alpha \cos \gamma - \cos \beta}{\sin \alpha \sin \gamma} \right) \quad (\text{B.6})$$

$$\gamma^* = \arccos \left(\frac{\cos \alpha \cos \beta - \cos \gamma}{\sin \alpha \sin \beta} \right) \quad (\text{B.7})$$

B.1.3 UB Matrix

The UB Matrix is used to convert from relative units to absolute units. It is made up of two matrices, one to scale the system, and one to rotate it.

The B Matrix scales the system and converts from a non-orthogonal reference fram to an orthoganol one.

$$|B| = \begin{pmatrix} A^*, & B^* \times \frac{\cos \alpha \cos \beta - \cos \gamma}{\sin \alpha \sin \beta}, & C^* \times \frac{\cos \gamma \cos \alpha - \cos \beta}{\sin \gamma \sin \beta} \\ 0, & B^* \times \sqrt{1 - \left(\frac{\cos \alpha \cos \beta - \cos \gamma}{\sin \alpha \sin \beta} \right)^2}, & -C^* \times \sqrt{1 - \left(\frac{\cos \gamma \cos \alpha - \cos \beta}{\sin \gamma \sin \beta} \right)^2} \times \cos \alpha \\ 0, & 0, & \frac{2\pi}{C} \end{pmatrix} \quad (\text{B.8})$$

The U Matrix rotates the system.

$$|U| = \begin{pmatrix} \vec{O}_1 \times |B| \\ ((\vec{O}_1 \times |B|) \times (\vec{O}_2 \times |B|)) \times (\vec{O}_1 \times |B|) \\ (\vec{O}_1 \times |B|) \times (\vec{O}_2 \times |B|) \end{pmatrix} \quad (\text{B.9})$$

$$|UB| = |U| \cdot |B| \quad (\text{B.10})$$

This means conversions can be performed as follows:

$$\text{Relative units} \rightarrow \times |UB| \rightarrow \text{Absolute units}. \quad (\text{B.11})$$

$$\text{Absolute units} \rightarrow \times |UB|^{-1} \rightarrow \text{Relative units}. \quad (\text{B.12})$$

B.2 Calculate Angles from Experiment Values

B.2.1 Translate \vec{Q} into reciprocal space

$$Q_{abs}^{\vec{}} = \vec{Q} \times |UB| \quad (\text{B.13})$$

B.2.2 Use inverse Bragg law to calculate a_1 & a_2

$$a_1 = \arcsin \left(\frac{\pi}{|\vec{K}_i| \cdot D_m} \right) \quad (\text{B.14})$$

$$a_2 = 2 \cdot a_1 \quad (\text{B.15})$$

B.2.3 Use cosine rule to calculate a_4

$$a_4 = S_s \arccos \left(\frac{|\vec{K}_i|^2 + |\vec{K}_f|^2 - |Q_{abs}^{\vec{}}|^2}{2|\vec{K}_i||\vec{K}_f|} \right) \quad (\text{B.16})$$

B.2.4 Calculate a_3 using vector triangle

$$a_3 = -\arctan 2 \left(\frac{Q_{abs_y}^{\vec{}}}{Q_{abs_x}^{\vec{}}} \right) - \arccos \left(- \left(\frac{|\vec{K}_f|^2 - |\vec{K}_i|^2 - |Q_{abs}^{\vec{}}|^2}{2|\vec{K}_i||\vec{K}_f|} \right) \right) \quad (\text{B.17})$$

B.2.5 Use inverse Bragg law to calculate a_5 & a_6

$$a_5 = S_a \arcsin \left(\frac{\pi}{|\vec{K}_f| \cdot D_a} \right) \quad (\text{B.18})$$

$$a_6 = 2 \cdot a_5 \quad (\text{B.19})$$

B.3 Calculate Experiment Values From Angles

B.3.1 Wavelengths

Using the Bragg law:

$$|\vec{K}_i| = \frac{\pi}{D_m \sin a_1} \quad (\text{B.20})$$

$$|\vec{K}_f| = \frac{\pi}{D_a \sin a_5} \quad (\text{B.21})$$

B.3.2 Calculating \vec{Q}

Using the cosine rule and the vector triangle:

$$|Q_{abs}^{\vec{}}| = \sqrt{|\vec{K}_i|^2 + |\vec{K}_f|^2 - 2|\vec{K}_i||\vec{K}_f| \cos a_4} \quad (\text{B.22})$$

$$Q_{abs_x}^{\vec{}} = |Q_{abs}^{\vec{}}| \cdot \cos \left(-\arccos \left(-\frac{|\vec{K}_f|^2 - |Q_{abs}^{\vec{}}|^2 - |\vec{K}_i|^2}{2 \cdot |Q_{abs}^{\vec{}}| \cdot |\vec{K}_i|} \right) - a_3 \right) \quad (\text{B.23})$$

$$Q_{abs_y}^{\vec{}} = |Q_{abs}^{\vec{}}| \cdot \sin \left(-\arccos \left(-\frac{|\vec{K}_f|^2 - |Q_{abs}^{\vec{}}|^2 - |\vec{K}_i|^2}{2 \cdot |Q_{abs}^{\vec{}}| \cdot |\vec{K}_i|} \right) - a_3 \right) \quad (\text{B.24})$$

$$\vec{Q} = Q_{abs}^{\vec{}} \cdot |UB| \quad (\text{B.25})$$

B.3.3 Calculating \vec{K}_i

$$\vec{K}_i = (|\vec{K}_i| \cdot \cos a_3, |\vec{K}_i| \cdot \sin a_3, 0) \quad (\text{B.26})$$

B.3.4 Calculating ΔE

$$\Delta E = 2.072 \cdot (|\vec{K}_i|^2 - |\vec{K}_f|^2) \quad (\text{B.27})$$

B.4 Plot Triple Axis Spectrometer from Angles and Lengths

Cartesian coordinate system.

$$M = (\text{const}, \text{const}) \quad (\text{B.28})$$

$$R = (M_x, +ve) \quad (\text{B.29})$$

$$S_x = (M_x + l_{ms} \sin(a_2)) \quad (\text{B.30})$$

$$S_y = (M_y + l_{ms} \cos(a_2)) \quad (\text{B.31})$$

$$A_x = (S_x + l_{sa} \sin(a_2 + a_4)) \quad (\text{B.32})$$

$$A_y = (S_y + l_{sa} \cos(a_2 + a_4)) \quad (\text{B.33})$$

$$D_x = (a_x + l_{ad} \sin(a_2 + a_4 + a_6)) \quad (\text{B.34})$$

$$D_y = (a_y + l_{ad} \cos(a_2 + a_4 + a_6)) \quad (\text{B.35})$$

B.5 Calculate Angles from the Positions of Spectrometer

$$a_2 = (\pi - \angle RMS) \quad (\text{B.36})$$

$$a_1 = \frac{1}{2}a_2 \quad (\text{B.37})$$

$$a_4 = (\pi - \angle MSA) \quad (\text{B.38})$$

$$a_6 = (\pi - \angle SAD) \quad (\text{B.39})$$

$$a_5 = \frac{1}{2}a_6 \quad (\text{B.40})$$

Appendix C

Submitted Files

This section aims to index the submitted content by providing a list of the submitted files and their locations.

C.1 Code

The code is organised in a hierarchical package structure and should not be altered manually. More information on the structure of the code is included in the javadoc documentation.

The code should be readily importable into an IDE such as eclipse, however some further configuration of the class path, for example, may be required.

The program is also supplied compiled as a platform independent jar file. This jar file can be used with the java webstart file, `vTAS.jnlp` to allow the program to be run over the internet. If recompiling the project to use with web start, the jar file will have to be re-signed.

C.2 Documentation

The documentation of the project takes the form of `html` javadoc documentation. This is located in the javadoc subdirectory, and can be accessed from `index.html`

C.3 Web Page

The program was designed to replace an applet, and therefore a replacement web page allowing the program to be run is included. The subdirectory `web/` contains everything needed to create this page, and the contents can be copied directly to a web server.

Index

analyser, 10

Bragg diffraction, 9
Bragg law, 10, 30

crystal system, 7

detector, 11
documentation, 20

existing program, 12

flat cone, 11

internationalisation, 19

javadoc, 20

load/save, 13

model-view-controller, 14
monochromator, 9

neutron spectrometry, 6
nuclear reactor, 9

reciprocal space, 29

sample, 10
source, 9
spallation, 9

triple axis spectrometer
 components, 9

UB Matrix, 29
UB matrix, 9
user interface, 15

visualisations

 calculation, 31
 model, 17

vTAS
 architecture, 14
 design, 14
 mathematical model, 17
 specification, 13

Bibliography

- [1] BARTHELMY, D. Unit cell dimensions. <http://webmineral.com/help/CellDimensions.shtml>.
- [2] BARUCHEL, J., HODEAU, J. L., LEHMANN, M. S., REGNARD, J. R., AND SCHLENKER, C. *Neutron and Synchrotron Radiation for Condensed Matter Studies*. EDP Sciences - Springer-Verlag, 1993.
- [3] BUSING, W. R., AND LEVY, H. A. Angle calculations for 3- and 4- circle x-ray and neutron diffractometers. *Acta Cryst.* (1966).
- [4] EGELSTAFF, P. A. *Thermal Neutron Scattering*. Atomic Energy Research Establishment, Harwell, Berkshire, England, 1965.
- [5] HOOKE, J. R., AND HALL, E. *Solid State Physics*. Wiley, 1995.
- [6] LE DELLIOU, N. Simulation d'un spectrometre trois-axes. (Previous Stàge's Report), 2006.
- [7] SUN-MICROSYSTEMS. Java documentation. <http://java.sun.com/j2se/1.4.2/docs/api/index.html>.
- [8] SUN-MICROSYSTEMS. Java internationalisation documentation. <http://java.sun.com/j2se/1.4.2/docs/api/java/util/ResourceBundle.html>.
- [9] SUN-MICROSYSTEMS. Java internationalisation tutorial. <http://java.sun.com/docs/books/tutorial/i18n/index.html>.
- [10] VARIOUS. Java forums. <http://forum.java.sun.com/>, 2007. (Many helpful clarifications, especially on JWS).
- [11] WIKIPEDIA. Wikipedia. <http://en.wikipedia.org/>, 2007. (Various entries - used as an introduction to several topics).

The author wishes to thank the staff of the ILL for their hospitality over the summer. In particular, acknowledgement must be given to Martin Boehm for his patient explanation of the physics and help with the endless debugging of the project, Alain Filhol for his tours of the reactor and invaluable clarifications, Arno Hiess for the feedback on the interface, and everyone who read over the report and highlighted the numerous mistakes.



Peter Braden
Summer 2007
<http://www.ill.fr/Computing/resources/software/vTAS/>
Institut Laue - Langevin
Grenoble
France