ICANS-XV
15th Meeting of the International Collaboration on Advanced Neutron Sources
November 6-9, 2000
Tsukuba, Japan

16.4
Towards a Responsive and Interactive Graphical User Interface For Neutron Data Reduction and Visualization

Alok Chatterjee[1*], T. Worlton[1], J. Hammonds[1], D. Mikkelson[2], R. Mikkelson[2], D. Chen[3] and C.-K. Loong[1]

[1] Argonne National Laboratory, Argonne, IL, USA
[2]University of Wisconsin-Stout, Menomonie, WI, USA
[3]Neutron Scattering Laboratory, China Institute of Atomic Energy, Beijing, China
[*] E-Mail: achatterjee@anl.gov

Abstract

An Integrated Spectral Analysis Workbench, **ISAW** has been developed at IPNS with the goal of providing a flexible and powerful tool to visualize and analyze neutron scattering time-of-flight data. The software, written in Java, is platform independent, object oriented and modular, making it easier to maintain and add features. The graphical user interface (GUI) for ISAW allows intuitive and interactive loading and manipulation of multiple spectra from different "runs". ISAW provides multiple displays of the spectra in a "Runfile" and most of the functions can be performed through the GUI menu bar as well as through command scripts. All displays are simultaneously updated when the data is changed using the Observable-observer object-model pattern. All displays are observers of the Dataset (observable) and respond to changes or selections in it simultaneously. A "tree" display of the spectra in run files is provided for a detailed view of detector elements and easy selection of spectra. The operations menu is instrument sensitive so that it displays the appropriate set of operators accordingly. Automatic menu generation is made possible by the ability of the DataSet objects to furnish a list of operations contained in the particular DataSet selected at the time the menu bar is accessed. The transformed and corrected data can be saved to a disk in different file formats for further analyses (e.g., GSAS for structure refinement).

1. Introduction

ISAW has been developed at the Intense Pulsed Neutron Source, Argonne National Laboratory, to provide a means to look at neutron scattering data (IPNS Run files). At IPNS, pulsed neutrons are used to characterize materials using time-of-flight [1] neutron scattering techniques. The raw data often must be preprocessed to subtract background noise, remove bad detector contributions, normalize by the incident spectrum and so on. ISAW provides the means to view and manipulate a large number of spectra interactively, therefore, the software should offer effective tools to display and discern anomalous data so that users can quickly tell if some detectors in the experiment are dead or noisy, but also can compare a series of spectra with respect to the variation of a physical parameter, for example, the temperature or pressure.

ISAW has been written in Java mainly because the software needed to be portable and platform independent. This was a primary requirement as IPNS has a vigorous user's program wherein visiting researchers who perform experiments at IPNS may want to analyze the data at their home institutes using a variety of commonly available platforms. Access to ISAW over the web was also one of the initial design considerations. In addition to being portable and platform independent, Java also provides a pure object-oriented framework making it very extensible. As a case in point, different parts of ISAW have been developed by individuals, some using MS-Windows and some using Linux. The effort to integrate these parts into the final product has been quite minimal. The absence of pointers makes Java comparatively easier to learn than C++ and also makes it less error-prone. Java provides a robust, fault-tolerant and secure framework to develop applications with features like exception handling, policy-based access control, certificate interfaces and X.509v3 implementation [2].

2. ISAW Organization

Most present day applications are developed using the Model-View-Controller (MVC) pattern [3, 4]. This pattern separates the application data (the model) from its presentation (the view). Furthermore the flow of the application based on user interaction (the controller) is also separated, see Fig.1. ISAW has been developed along the MVC pattern and thus can be conceptually split into three tiers. The first tier consists of the graphical user interface classes that are present in a package called IsawGUI. Swing class libraries that are provided as a part of the Java Foundation classes and contain components such as frames panels, buttons, etc. have been used in IsawGUI to build the presentation interfaces. The second tier consists of the controller. These classes, present in the Swing libraries, respond to user events and provide data to IsawGUI. The communication between the GUI and the controller is carried out using standard Java event mechanism. The third tier consists of the model or the application data. In ISAW, the model is a DataSet object (a collection of spectra together with associated information such as attributes, operations, axis units, etc.) which knows nothing about how the data are presented to the user. In this article we discuss the first tier, the graphical user interface. The Data model will be discussed in another article.

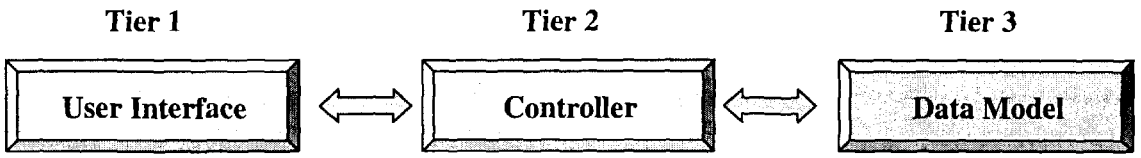


Figure 1. The MVC architecture

3. IsawGUI layout & features

The look and feel of an application is a critical factor in determining its ultimate value to the end user. The interface provided by ISAW is designed to be highly usable and can be easily learned resulting in enhanced user productivity. When ISAW is launched it displays a screen that is divided into four parts (Fig. 1). The top left panel referred to as the "tree" display is where the files structure is displayed in tree form. The top right panel referred to as the "view" display shows the different views that can be invoked on the run files. The bottom left panel referred to as the "attribute" display shows the properties or attributes of objects that are selected in the "tree" display. Finally, the bottom right panel referred to as the "command"

display has a number of tabbed panels that display the history of the operations performed on the selected run file, the "Session Log", some system related information and an area in which to store and invoke scripts. In addition to the four display panels there is a menu bar with various menus and menu items. The menu items along with the "scripts" are the primary mechanisms by which various functions, operations and viewers can be invoked on the data.

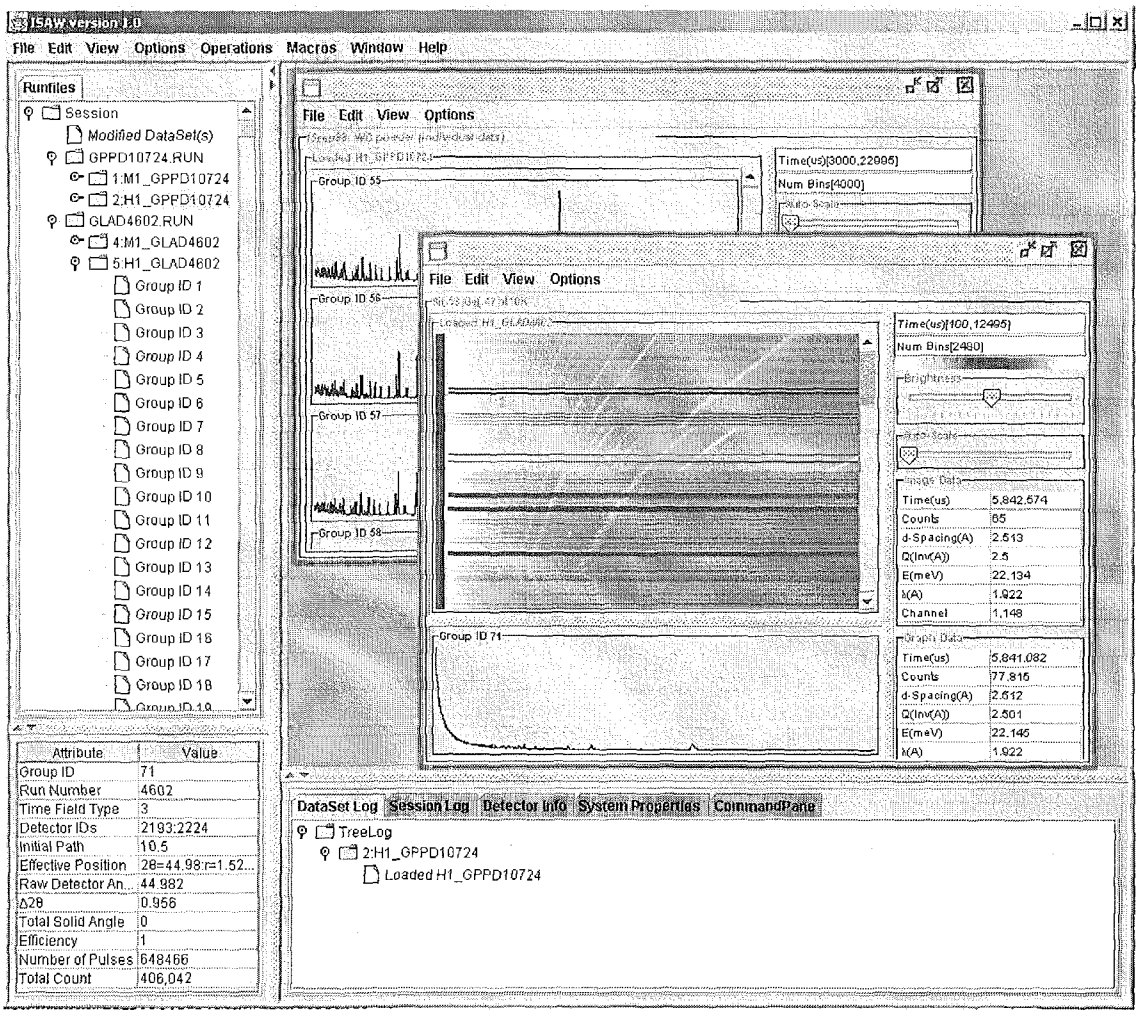


Figure 1. Snapshot of the IsawGUI window displaying a loaded run file along with two different viewers.

ISAW menu bar

The menu bar consists of the File, Edit, View, Options, Operations, Macros, Window and Help menus as shown in Fig 2. Each menu provides a list of menu items corresponding to various tasks. For instance, the File menu provides various ways to load run files. A file dialog box allows user to select files over the local area network.

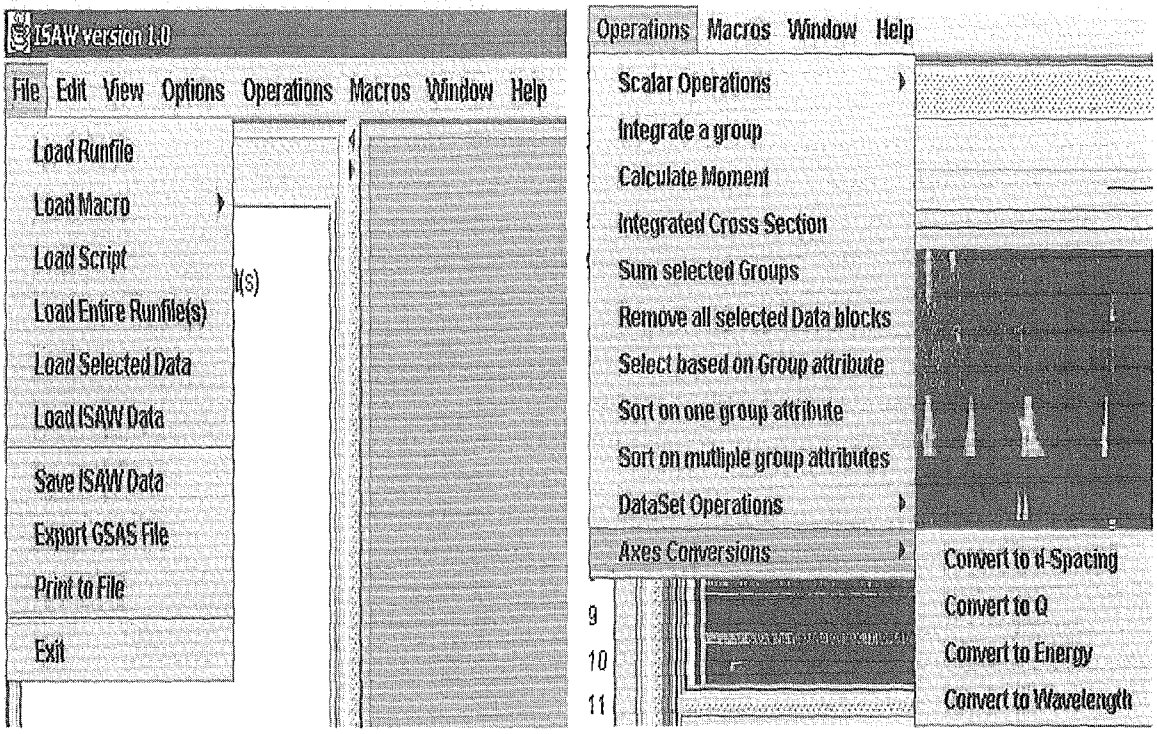


Figure 2. The ISAW menu bar displaying the File and the Operations menu items

The "Operations" menu allows users to invoke different operators that combine and modify one or more data sets and produce a new data set. Basic operations such as scalar addition, multiplication, integrated cross section etc. are supported by all DataSets. Each new DataSet for an instrument carries with it a list of relevant "operators" based on the instrument type. When a new DataSet is created by a DataSetFactory object, the appropriate operators are assigned to the DataSet. The Operations menu is automatically populated with the names of the operator objects included in the currently selected DataSet. Each operator object carries with it the list of required parameters. A generic parameters input dialog box is displayed when an operation is invoked and allows users to specify the parameters needed by the operation. This allows new operators to be written and used without any changes to the user interface.

"Tree" display:

The "tree" display shows the loaded runfile by putting it into a "tree node" after separation into monitor and histogram DataSets. M and H prefixes are used to distinguish the monitor and the histogram data, Fig.3a.

"Attribute" display:

Each DataSet and Data (individual spectra) object contains a list of attributes. The list of attributes associated with the currently selected DataSet or Data is displayed in the "attribute" display area as shown in Fig. 3b.

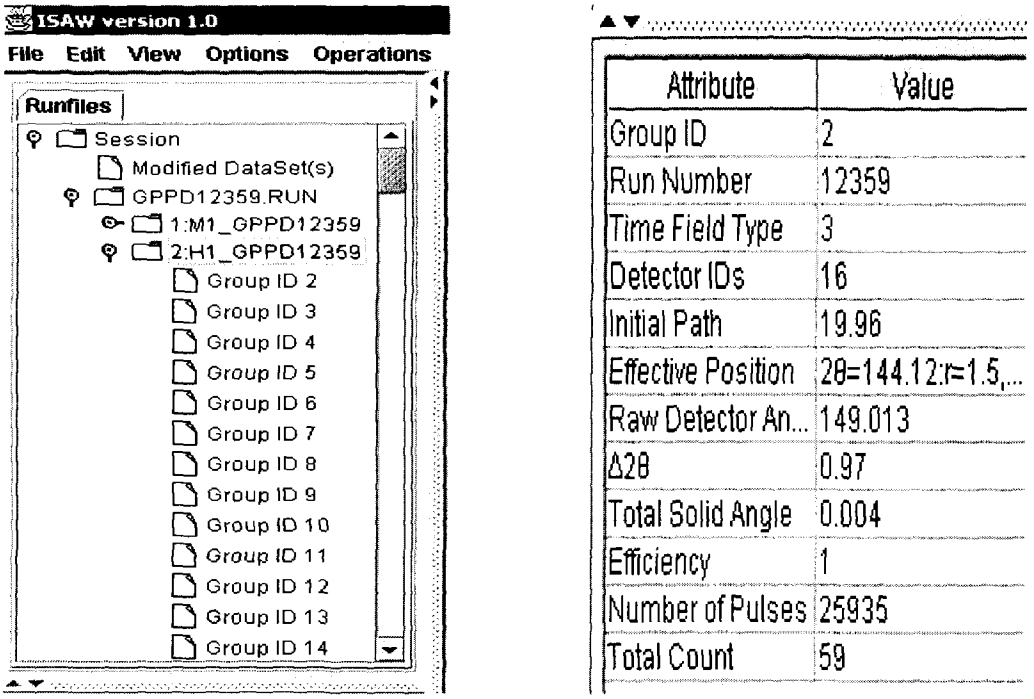


Figure 3 (a) A selected data item in the "tree" display and (b) the corresponding attribute values in the "attribute" display.

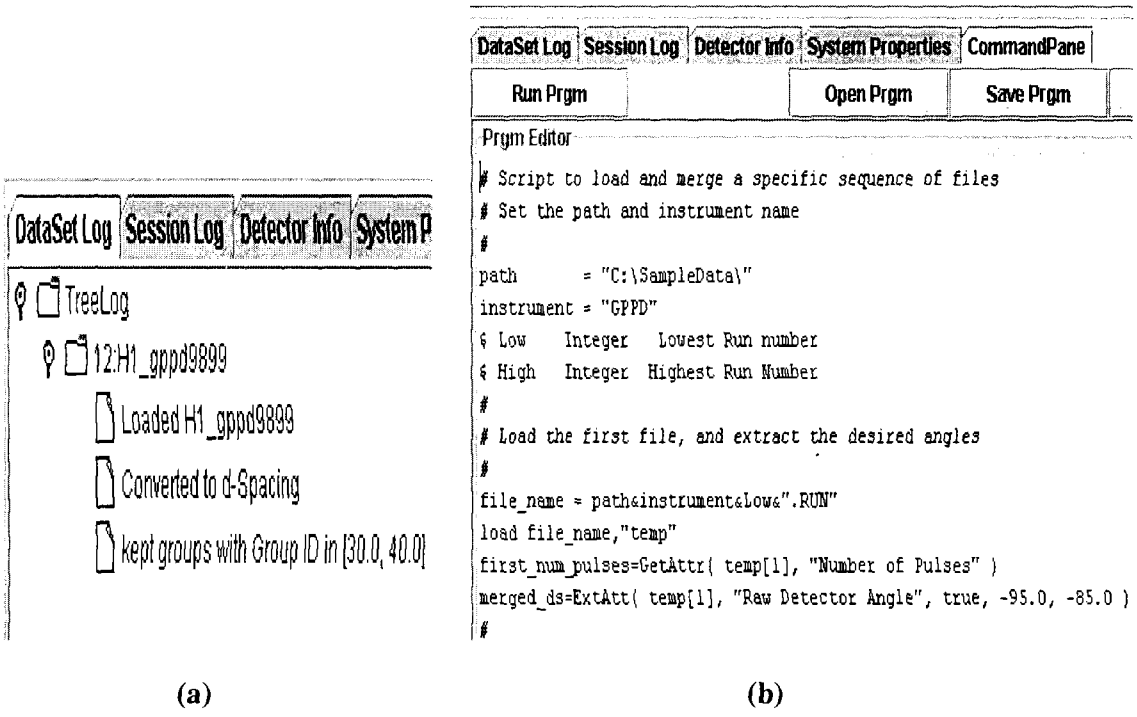
"View" display:

Several views of the DataSets are currently supported (see Fig 1.). These views appear "internally" inside the "view" display area. They can also be generated external to the IsawGUI window. The different views include an image display of the all the spectra in a DataSet and also a scrolled set of line graphs. The images and graphs can be zoomed in/out to magnify/reduce the detail features. The image and graphs view also provide various ways to visualize and manipulate the data by allowing multiple spectra selection and performing deletion, sorting, etc. on the selected spectra.

"Command" display:

The "command" display area provides a number of tabbed panels that convey different kinds of information to the user Fig. 4 (a, b). The DataSet Log displays the log of operations for the currently selected DataSet. The Session Log displays all actions including operations that are performed during an ISAW session. Detector Info provides some information on the detectors in a group and System properties tells about the memory and system parameters. The CommandPane allows users to write scripts, save scripts and open previously saved scripts. Each operator has been provided with an "easy to use" textual name and can be used in scripts. The CommandPane class provides the following:

1. A text editor pane to compose and test command scripts.
2. An immediate pane to execute one instruction at a time.
3. Automatically generated dialog boxes for parameters to scripts.
4. Transfer of DataSets to/from an observer program.
5. Execution method for command scripts passed in from an observer program.
6. Batch mode execution of command scripts.



(a) (b)

Figure 4 (a) Snapshot of the DataSet Operations log (b) Partial view of a script in the CommandPane.

4. Component updates: Design by Notification

When the selection of the DataSet object changes in the "tree" display it is important to have a consistent and true view of the currently selected DataSet in the other displays of IsawGUI. Thus, some mechanism to track the currently selected DataSet object needs to be established. We have used a form of active notification, wherein the notification responsibility lies with the object that changes. This object-model pattern [5] is also known as the Observable-Observer model, Fig 5. In this model the Observable maintains a list of Observers. When the Observable object changes in a significant way, it notifies each of its observers. After the Observers receive the notification from the Observable they need to take appropriate actions to "update" themselves. This model has two main advantages:

1. It reduces overall message traffic.
2. It reduces the coupling that an observable has with its observers.

In ISAW a DataSet object is an observable that carries with it a list of observers, such as the different views, the "tree" display, the "attribute" display and so on. As the currently selected DataSet is changed either by invoking some operation on it or by merely selecting a different DataSet (by Message Inward) the "observers" are all notified and updated by calls to their "update" methods.

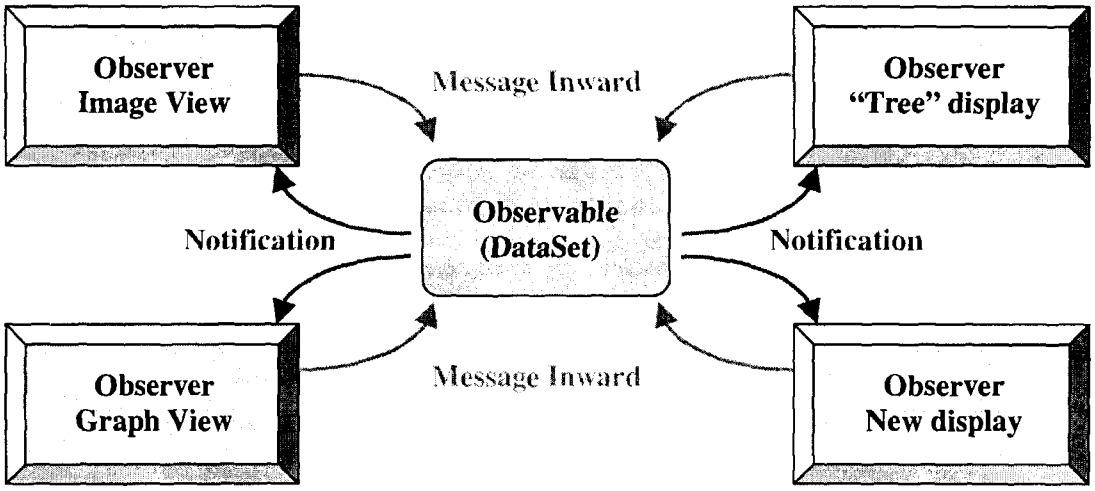


Figure 5. Schematic of the "Observable-Observer" object-model pattern

5. Summary

A highly interactive, portable, and extensible interface, IsawGUI, for spectral analysis of neutron scattering data has been developed. IsawGUI tries to provide a consistent look and feel for ease of use. It is designed to minimize the coupling between the data model and presentation layer. The software is being refined and expanded to support more user case scenarios and it is anticipated that very soon it would progress from the 1.0 beta version to a full release.

References

[1] C. G. Windsor. 1981. Pulsed Neutron Scattering. London: Taylor and Francis
[2] http://www.w3.org/
[3] http://java.sun.com/products/jfc/
[4] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Design Patterns.
[5] Peter Coad, Mark Mayfield, Java Design, Yourdon Press Computing Series.