Quotation for XYZ, a Quality Control Tool

Prepared for: The US Geological Survey

September 1, 2006

1 General Information

Instrumental Software Technologies, Inc. (ISTI) is pleased to submit this quotation to provide software development to The USGS for the XYZ data quality control tool. ISTI has written several programs of this nature for seismology and is well suited to develop the XYZ tool. ISTI has been developing data processing applications for academia and governmental organizations since its inception in 1998 and prides itself on leveraging existing open source technologies to provide robust, cost effective and long lasting solutions.

The requirements of the current RFP, such as object-oriented modularity, portability, testability and extensive are applicable to software projects previously developed by ISTI. Our mission is to provide professional-grade software and services to the scientific geophysical community in the US and worldwide. ISTI is, and has been since its inception, an open source software development company.

We develop custom software — customer satisfaction is the main priority of our work.

ISTI differentiates itself from a standard geophysical software contractor in that software development at ISTI is a team project. Multiple developers in the company share the software internals. This guarantees that ISTI will be able to support the software many years in the future.

ISTI differentiates itself from a large software company in that ISTI consists of a dedicated group of talented, highly motivated engineers. Support calls and emails are handled by the developers themselves and are usually answered within an hour. Problems are fixed within days.

Since it's inception, ISTI has successfully completed dozens of projects compatible in size and scope to the current RFP. See our website at www.isti.com for a list of completed projects and current customers.

2 Technical Proposal

In this section we outline our approach to software development in general and the proposed technical solutions for this specific project.

2.1 Software Development Process

ISTI follows a classical 4-phase software engineering development cycle of “specify, design, implement, and test”. That is, we first make sure that the specifications of the customer are well described and reasonable from an engineering standpoint, and then we develop a design document with mock-ups of the application. After the customer has vetted the design document we begin an implementation phase where we develop the modules and finally we test the modules (with unit tests as possible) before providing iterative releases to the customer to review. Before the final release we work with the customer to assure that all requirements have been met by providing multiple candidates for release. Once a release is complete, we provide documentation as a final phase of development.

Within this development process, ISTI will be working with the USGS personnel in Golden and Albuquerque, as required in the **Task Description** in the software specification in the Request for Quote.

ISTI will also deliver according to the define **Milestone** schedule in the software specification in the Request for Quote.

2.2 Proposed Technical Approach

2.2.1 Requirements Acceptance

ISTI has reviewed the XYZ specifications document and can meet all of the requirements stated therein. Any needed clarifications to the specifications document are noted below.

2.2.2 Programming Language

ISTI proposes to use the Java programming language to develop the XYZ tool. The minimum version of Java supported by this project will be 1.4. Java provides a platform independent execution mechanism using the Java Virtual Machine, which runs on all of the specified operating systems. ISTI has been developing with Java since its inception and has 4 developers well versed in its use. Furthermore, ISTI has a large toolkit of modules that we can provide to speed the development of the tool. The ISTI Util toolkit includes configuration and command line utilities, logging support, and other often used widgets and libraries that ISTI has found useful over the last 8 years.

Java is a modern object oriented and multithreaded language that allows for modular development. All tasks within the modules will be written as discrete Java classes and hence be reusable in future projects. The modules can be tightly or loosely coupled depending on the use, but will involve separate archive files (“jars” in Java jargon) and packages within the jars that perform specific tasks and functions on the data.

2.2.3 Modular Design

ISTI proposes to develop this software as a series of modules; for both processor engines and for the graphical user interface (GUI) front-end. The GUI front-end will be coupled to the processor engines, such that processing results can be achieved in a multithreaded manner when necessary.

Based upon review of the RFP document, ISTI plans on designing a system that is data/data stream driven. Data will be manipulated based upon the standard NSCL (Network/Station/Channel/Location) organization model, allows for different data streams to be grouped arbitrarily and processed according to these user definable groupings.

With the data manipulation structure defined, the data objects may be populated from any data source. So processing will be independent of access method and data format. Having the system work with either “In the Clear” data or multiplexed data retrieved in near real time from remote data sources will be relatively transparent to the end user.

In the software specification there are many processing and analysis tasks that are defined. Each of these specific tasks will have a distinct mode. Each mode may have either distinct data products, or data visualization tools (or both).

As is indicated in the specification, there will be a series of modes that may be enabled and disabled according to the needs of the user. Each mode will be associated with the data currently on display, and will modify the user interface as needed. For example, a mode that allows for filtering the data will allow the user to specify the filter parameters, and will popup input and output windows as needed. A mode may also modify the default mouse and keyboard behavior on an as needed basis. ISTI will work closely with the USGS to specify how the various modes are enabled/disabled in order to ensure that control of the software is intuitive. And for all of the GUI’s ISTI will work with the USGS to ensure within a single mode that they user interface is also intuitive.

2.3 Technical Specification

After reviewing the technical specification, ISTI has some preliminary thoughts on the design and implementation of the XYZ software. These initial design ideas will be discussed in great detail with the USGS during the design phase of the project, however we present our initial ideas here in order for the USGS to properly evaluate the validity of our design.

2.3.1 QC Gui Functional Requirements

In the discussion below, when existing tools and toolkits are listed in **bold**, this means that ISTI plans on using these existing tools in this project. If a tool is mentioned as an example of ISTI's experience and familiarity with a task, it is listed in *italics.*

1. Inputs
    a. Equally sampled timeseries in miniSEED format – ISTI has **existing modules** written in both Java and C for reading miniSEED data.
        i. In the Clear data – ISTI has **existing modules** for reading clear data as either ints or floats, written in both Java and C.
    b. Unequally Sampled timeseries of parameters in XML. – ISTI has experience with reading, displaying and processing data using unequally sampled data (i.e. PASSCAL *Logviewer*, Various tools for performing timing corrections on data). ISTI also has experience reading and writing XML data. ISTI uses the open source **JDOM Java tools for XML** parsing, which we have extended as ISTI specific classes for additional functionality.
    c. See b)
    d. Metadata in dataless SEED format – ISTI has written tools in Java (**SHAPE**) and contributed to community Java tools (**DMC's JavaSeed**) to write and read dataless SEED volumes.
    e. Data Access – In general ISTI has extensive experience with Data Access, ranging from local file access (*miniSEED, etc)*, to remote socket based server access (*waveserverV, DHI, etc)*, to direct instrument access (*Most seismic hardware)*
2. Outputs – In general ISTI has extensive experience with writing data and metadata to external formats. For this section ISTI will reuse existing open source solutions as much as possible, and write new open source code as needed for tasks not currently implemented.
    a. Timeseries in miniSEED and XML. – ISTI has some **existing tools** for writing miniSEED files and writing XML files. Some additional software and specific XML formatting will be required for this task.
    b. Non-timeseries In the Clear data - ISTI will create output writers as needed for this task.
    c. Panel or pop-window images in standard graphics format. While ISTI has experience with all of the listed output formats, ISTI prefers to use PNG images for output. PNG is an open standard and well supported in Java.

d. Updated/Revised QC reports in HTML. New code will be written to support this task.

e. Updated flow control information via an API to a database. ISTI has experience with database interactions from software. Databases used by ISTI include (Oracle, Sybase, MySQL, PostgeSQL, etc).

f. Updated metadata in RESP format. ISTI is the author of **SHAPE**, a tool for parsing and creating RESP formatted files and integrating them with Dataless SEED volumes. SHAPE was developed for ORFEUS and IRIS, and is currently supported by ISTI for the USGS's use within the INV Inventory Management Software.

3. GUI Characteristics – Much of the new work on this project will be in support of the GUI Elements. Unless mentioned explicitly below, ISTI will be writing custom or extending existing code for these tasks.

   NOTE: ISTI proposes to have an optionally displayed legend panel/window that will show the user a number of useful runtime settings. For example, in many of the modes described below the mouse behavior will change from mode to mode, and panel to panel. Also, colors and shapes will be used in this software to indicate many different things (i.e. an event for which synthetics exist). This runtime window will have a mouse hints graphic, and an active legend to help provide context to the windows.

   a. Start-up
      i. Read configuration files. – Standard software implementation which is available in the ISTI Java Util class that will be provided.
      ii. Accept user input on the command line and accept wildcards. - Standard software implementation which is available in the ISTI Java Util class that will be provided.
      iii. Minimal start-up specification. –ISTI will support the standard defaults. The system may optionally be configured to remember the settings used in previous runs.
      iv. Load optional reports for selected stations/channels. – this will be an extension of normal report loading.

   b. Display and Processing –
      i. Display behavior – All display panels will function independently, and may be assembled in any way. Because of this, we could have an application that has all panels appear in a single window, or each panel in its own window. The software will function the same in either case. Because of this, ISTI will be able to change the arrangement of the panels at any time during development if the USGS wishes. As an example, one piece of a Java software that ISTI has written had 3 display modes. The

first was a single master window with 3 free-floating panels within that window. The second was a single master window with 3 tabbed panes allowing easy selection of the sub panels, and the third display mode was with 3 freely floating windows with no master window. The processing code functioned identically in all cases.

ISTI has **existing software objects in its Java toolkit** for displaying seismic traces. This software allows for setting the display to arbitrary time windows and amplitude scale via software, and via user input. The individual panels may be overlaid upon each other. Trace Clipping is the desired behavior described in the specifications document, and is the default behavior in the ISTI trace panel, however it is also possible to have the traces overlap rather than clip if desired. Also, each panel can hold a collection of other panels instead of (or in addition to) the data. This allows a single panel to hold a panel of the raw data, a panel of arrivals, a panel of data gap information, a panel holding a synthetic seismogram, etc. and all will be displayed correctly. Also, each panel has a depth defined so that they are layered correctly when displayed. Each panel supports data value inquiries, via interactions between the panel pixel information and the underlying data object.

In support of this project, ISTI will need to extend these panels to add or modify the base behavior. But these tools have proven themselves to be easily extendable in the past.

ii. Auto QC/Event files

1. Upon loading the AutoQC/event files, GUI will be modified according to the spec's requirements
2. AutoQC information will be displayed and be user selectable. The GUI will be modified based upon the user selection and in accordance with the specification. Issues will be displayed according to a configurable priority order. If the USGS would like ISTI to support having an Issue declare its own priority, we can support that also.
3. Event Information –
    a. Display - Events will be displayed according to the software specification.
    b. Event representation - This, as with all windows and panels, will make use of our proposed Legend Panel for explaining the use of colors and symbols (i.e. indicating which events have synthetics associated with them.)

c. Channel specific event information will be associated with all appropriate trace windows.

d. User Actions – in the list of user actions (i – x in the specifications document) all specifications will be supported. Some of these tasks will require new data products to be associated with existing data display panels while other will change how the existing data display panels "display themselves" (and others will do both). Where data display is concerned, ISTI's data reading tools can be configured to retrieve data automatically when necessary. In its simplest form this may be used when a display needs to be adjusted forward in time and data is not currently loaded into memory. In a more complex form, event data that is channel specific could actually load up channels that have not yet been loaded into memory.

The GUI will be able to view and update QC reports.

Zooming is currently allowed at any scale for ISTI's trace panel. If necessary, ISTI will support multiple levels of memorized zooming, as per item 3.b.ii.3.d.vii. As zooming is a function of the panel, the mouse or any piece of the software may request that the panel's scale increase by any factor. The data section affected will then tell the panel how to redraw the trace.

Pop-Up windows will be used as per the software specification. As mentioned earlier, Pop-Up windows may also be used as part of the main window if desired.

The ability to view the seed headers and generate maps as add-ons will be possible.

Data Processing – From the perspective of XYZ, performing tasks such as filtering, cross-correlation, are processing tasks, or actions. An object-oriented method of defining actions, what data they operate upon, and how they display their results will be designed. ISTI will develop a base class (called Action, possibly). Each implementation of an Action will need to have defined a data object that it acts upon, how it does its processing, and how the processing results are to be used/displayed/saved/etc. Possible results of an action may be adding data to an existing panel, interacting with a database, creating a new pop-up panel, or generating output.

The system will be designed in such a way that anyone familiar with Java will be able to create his or her own Action objects. As long as new Actions conform to the Action interface, new user actions may easily be incorporated into the XYZ tool in the future.

All output would be able to be saved as per the software specification.

c. Metadata – As per the specification. Metadata will be viewable and editable as needed.
d. Closeout – The software will generate QC reports and update the flow control as needed.

4. Other
   a. As ISTI is an open source software development company, all components developed for this project will be open source. In addition, most (if not all) additional components will also be open source.
   b. The GUI shall run and be tested by ISTI on the listed platforms (Solaris, Linux, OS X).
   c. ISTI will make no assumptions with respect to screen type, other than a color display will be assumed. Geometry of the screen will be adjustable at run time. Due to the nature of using standard menus, a minimum screen height should be defined. If this is not possible, instead of using menus, we will need to use scrollable pop-up windows instead of pull down menus. This is easy to implement, but most users will expect normal pull down menus. Therefore normal menus (and defining a minimum screen size) may be preferable to the USGS.

2.4 Technical Requirements

ISTI will ensure that the program meets the need be able to display 10 days worth of three channels of 40 sps and three channels of 1 sps data, all in “in the clear” format within 10-12 seconds on a specified computer system.

The panel-based system described in this proposal supports all of the display requirements. The usability of a display with very small amounts of real estate per trace will vary from user to user. But the software will not prevent the user from using the display in this manner.

2.5 Documentation

ISTI diligently documents all of the Java classes and packages during development using the built-in JavaDoc tools. For an example of our use of JavaDoc, refer to the QuakeWatch Server documentation at this URL: http://www.isti2.com/QWIDS/QWIDS_dist_v1.5/QWLibs/javadocs

2.6 Schedule for Delivery

The Delivery Schedule required in the software specification will be adhered to.

2.7 Deliverables and Payment

ISTI will invoice the USGS monthly for each of the tasks outlined above as they are completed. ISTI intends upon providing the USGS with incremental releases to that new features may be tested as they are released.

2.8 Warranty

ISTI will fix any bugs in the software that are a result of our work. This does not include version changes in software libraries or virtual machines that are used or to modifications that are not made by ISTI.com.

ISTI’s standard software warranty is for one year. Support calls and emails are usually answered within an hour from arrival. Problems are fixed within days. ISTI is proud with our very fast support turn-around. Beyond this warranty, ISTI provides support on hourly basis. 24/7 support is also available.

2.9 End User Documentation

While it was not requested, ISTI recommends that end user documentation should be created for the final software product. This documentation can be written by the client, or ISTI may be contracted to write this documentation as a separate task