KNOWLEDGE BASE NAVIGATOR FACILITATING REGIONAL ANALYSIS INTER-TOOL COMMUNICATION

B. John Merchant, Darren Hart, Eric P. Chael, Matthew N. Chown, and Jeff W. Hampton

Sandia National Laboratories

Sponsored by National Nuclear Security Administration
Office of Nonproliferation Research and Engineering
Office of Defense Nuclear Nonproliferation

Contract No. DE-AC04-94AL85000

ABSTRACT

To make use of some portions of the National Nuclear Security Administration (NNSA) Knowledge Base (KB) for which no current operational monitoring applications were available, Sandia National Laboratories have developed a set of prototype regional analysis tools (MatSeis, EventID Tool, CodaMag Tool, PhaseMatch Tool, Dendro Tool, Infra Tool, etc.), and we continue to maintain and improve these. Individually, these tools have proven effective in addressing specific monitoring tasks, but collectively their number and variety tend to overwhelm KB users, so we developed another application – the KB Navigator -- to launch the tools and facilitate their use for real monitoring tasks. The KB Navigator is a flexible, extensible java application that includes a browser for KB data content, as well as support to launch any of the regional analysis tools. In this paper, we will discuss the latest versions of KB Navigator and the regional analysis tools, with special emphasis on the new overarching inter-tool communication methodology that we have developed to make the KB Navigator and the tools function together seamlessly. We use a peer-to-peer communication model, which allows any tool to communicate with any other. The messages themselves are passed as serialized XML, and the conversion from Java to XML (and vice versa) is done using Java Architecture for XML Binding (JAXB).

OBJECTIVE

MatSeis and the regional analysis tools continue to provide an excellent prototyping environment in which promising seismic analysis techniques can be implemented and evaluated quickly with relatively little effort. These products are now all fairly mature, but development on them continues in response to the research results from our co-workers at LANL and LLNL.

The KB Navigator is a relatively recent addition to the suite of tools that Sandia has developed. The goal of the KB Navigator has been to provide a simple, consistent way to access the tools and content of the KB. Another part of using the KB that is being addressed is the level of interoperability between the diverse set of tools. The Peer to Peer Tool Communication provides a flexible framework in which applications can interact with one another.

RESEARCH ACCOMPLISHED

The following sections describe the latest versions of KB Navigator, MatSeis, the regional seismic analysis tools, and the new tool communication capabilities.

KB Navigator

The KB Navigator is a graphical interface into the KB that allows a user to find the research products, check geographic extents, access standard GIS datasets, view metadata and launch the research tools. An Oracle database schema was designed for the KB Navigator to hold all the pertinent information about the products.

Nodes in the KB Navigator are created for each dataset. There are two distinct kinds of nodes: folder nodes and dataset nodes. Folder nodes provide an organizational hierarchy and can contain other folder nodes as well as dataset nodes. Dataset nodes represent a variety of files, dataset types, database queries (SQL) and research tools - all the contents of the KB as best as can be captured with one tool. Each dataset node includes as much descriptive information as possible: proper name and version, location of the data on disk, dataset type, descriptive keyword information and the tools that can be used for access.

The new version of KB Navigator includes a streamlined database interface that simplifies the process of importing and exporting database content by allowing the user to drag and drop the pieces of the hierarchy. Double clicking on a dataset entry in the browser will automatically execute the default tool for that dataset. Keyword entry has also been simplified by allowing the user to drag and drop keywords, delete keywords, and select from existing keywords using a pulldown list.

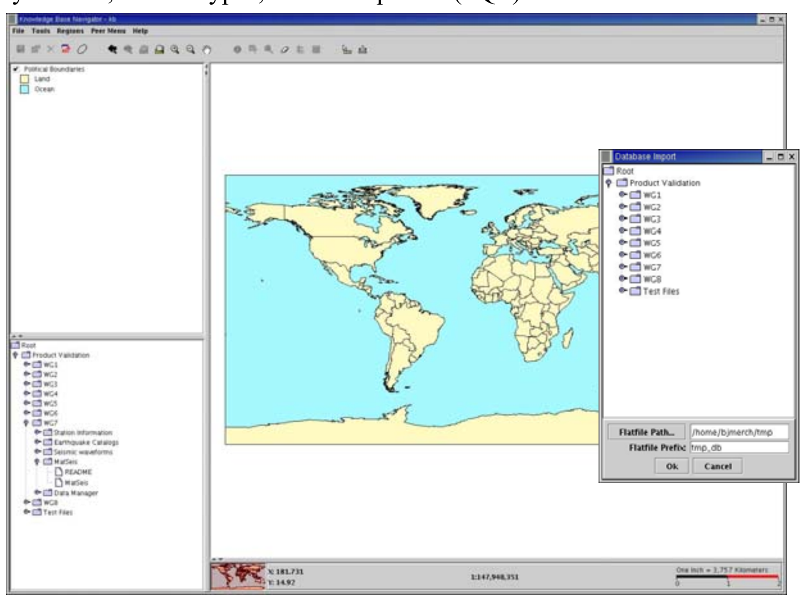

Figure 1. The KB Navigator and the Database Import interface

MatSeis 1.8

The main MatSeis graphical window is a standard time vs. epicentral distance plot that can display waveforms, arrivals, origins, and travel time curves. The user can interact with this display by clicking directly on the displayed objects, by using the buttons along the bottom, by using the menus along the top, or by typing commands at the Matlab prompt. MatSeis is predominantly written as Matlab m-file functions, which are organized in a set of directories according to the general purpose of each. However, the package also includes a set of compiled C functions and Java objects. Typically the compiled code is introduced where performance of an m-file is too slow (e.g. FK calculations) or cumbersome (e.g. managing the waveform, arrival, origin, and travel time objects). New features in MatSeis.1.8 are described below.

Support for System Environmental Variables
MatSeis 1.8 supports system environment variables within its configuration files. For example, in the following lines $MATSEIS_HOME will be replaced by the content of the environment variable MATSEIS_HOME:

```
setenv    MS_MAP_DATA    $MATSEIS_HOME/data/map
setenv    MS_MAP_FILE    $MATSEIS_HOME/data/map/world_15min.mf
```

Theoretical vs. Observed Travel Times
MatSeis 1.8 provides an interface for displaying actual arrival time versus theoretical arrival time. For a set of origins that have associated arrivals, we plot the relative arrival time versus epicentral distance, along with the theoretical travel time model data. Researchers can then see how their travel time model fits the observed arrival data.

Database Access
MatSeis was originally developed to read data from an Oracle SQL database using a command line utility to output the database results to a file on disk. This method of reading from a database was very inflexible and provided limited cross-database and cross-platform functionality. In order to make the software more generally useful and to provide access to any SQL Relational Database, for MatSeis 1.8 we have moved to using Java DataBase Connectivity (JDBC). This capability will allow MatSeis to access any type of database so long as a JDBC driver is available, and this makes it possible to read data from a remote database on the internet.

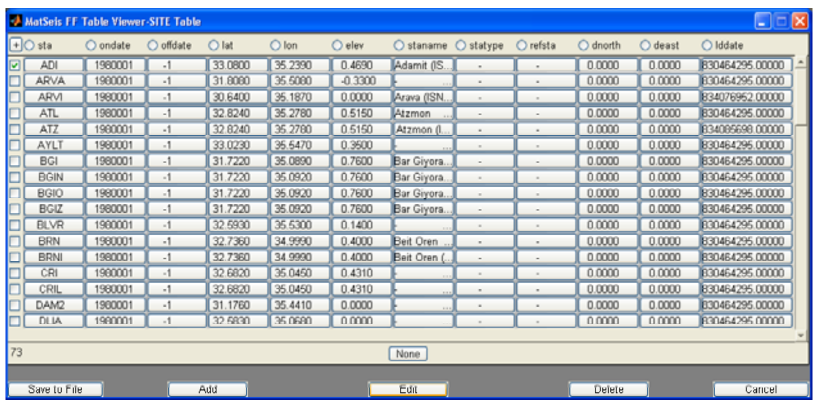

Figure 2. MatSeis Flatfile Table viewer

As part of our support for flatfile databases, CSS 3.0 and NNSA KB core table schema (Carr, 2002), we have added the ability to view flatfile database table contents by selecting from a menu within MatSeis (i.e. *View → CSS FF table Viewer → 'table name'*). The menu selection allows a user to choose from a subset of the available tables within the core schema. Current supported tables include affiliation, arrival, assoc, event, gregion, instrument, lastid, netmag, network, origerr, origin, remark, sensor, site, sitechan, sregion, stamag, and wfdisc. The viewing functionality was added to allow the user to verify the integrity of the flatfiles (error messages are displayed if the files have errors). In addition, the flatfile viewers provide the user with the ability to quickly read in select portions of a flatfile database.

In version 1.9 of MatSeis, users will also have the ability to add, edit or delete rows using this interface for their flatfile tables.

Signal Processing
MatSeis 1.8 contains several enhancements to simplify various signal processing operations. First, the FK interface now has the capability for the user to seamlessly transfer the azimuth and slowness information from the FK plot to the arrivals in MatSeis. Also, an interface has been added to allow the user to transfer arrivals from one element of an array to another by maintaining a time offset from the theoretical arrival time. Finally, the spectrogram and power spectral density plots can be animated to display the change in frequency content over time.

What's Coming in 1.9
MatSeis 1.9, due to be released in November 2004, will contain a number of useful feature enhancements. MatSeis will be able to write out wfdisc and binary waveforms data files. Two new MatSeis data types are being added for network and arrival magnitude association: netmag and stamag. These new data types are used in the same fashion as the existing MatSeis functions: arrival, origin, travtime and waveform. When reading in origins, if the netmag table exists and holds network magnitude values for the supplied origins, then those entries are read into netmag. In

a similar fashion, when arrivals are read into MatSeis, corresponding entries in the stamag table are read into stamag.

Phase Match Tool 1.8

PhaseMatch Tool is a waveform analysis interface launched from MatSeis that allows the user to calculate the predicted surface wave dispersion for a given source to receiver path by ray tracing through a model, and then use the model dispersion to generate and apply a matched filter (Herrin and Goforth, 1977). The tool allows the user to view the observed waveform, the model dispersion, the predicted waveform, the cross-correlation of the predicted and observed waveforms, and the match-filtered waveform. The user can control the frequency range of the model dispersion used, as well as the time limit of the portion of the cross-correlated waveform from which the match-filtered waveform is taken. Once a satisfactory filtering has been achieved, the user can send either the observed waveform or the filtered waveform to Measure Tool to measure surface wave amplitudes, which can then be used to determine event magnitude.

Phase Match Tool does not contain any new features for versions 1.8 or 1.9. However, beginning with MatSeis 1.9, Phase Match Tool will be included with the basic MatSeis package that is openly available on the GNEM R&E website.

Coda Magnitude Tool 1.8

CodaMag Tool is a waveform analysis interface launched from MatSeis that allows the user to calculate magnitudes and source spectra for an event of interest by fitting empirical decay functions to narrow-band coda envelopes of a given phase (currently Lg). The technique was developed by Mayeda and has been described in detail in several papers (Mayeda, 1993; Mayeda and Walter, 1996; Mayeda, et al., 1999). The tool consists of two displays. The main one shows the calculated moment spectrum and the derived magnitudes. The second display shows how the spectrum was derived. The user can adjust the Lg arrival window, examine the fit between the observed and synthetic envelopes, and control which frequency bands are used for the magnitude calculations. The various required parameters (frequency bands, groups velocity windows, decay curves, etc.) are read from parameter files unique to each station.

The only significant change for Version 1.8 is support for 2D tomographic Q models.

Event Identification Tool 1.8

EventID Tool is a waveform analysis interface launched from MatSeis that allows the user to identify an event of interest (i.e. explosion or earthquake) using spectral ratios of standard regional arrivals (see Hartse *et al*., 1997; Walter et al., 1999). The tool consists of three displays. The main display plots the phase ratio for the current event against a backdrop of the same ratio for archived events that have already been identified. The user can choose different phases and/or frequency bands to ratio to try to improve the separation of the earthquake and explosion populations, and the display will immediately update. A second display shows the user a plot of an "MDACogram" (i.e., the MDAC corrected measurements at all of the phase/frequency combinations) for the current event along with all of the archived events. This can be useful in deciding which ratio will yield the best separation. If there are questions about the amplitude measurements themselves, a third display can be brought up, and the user can easily examine group velocity windows for the

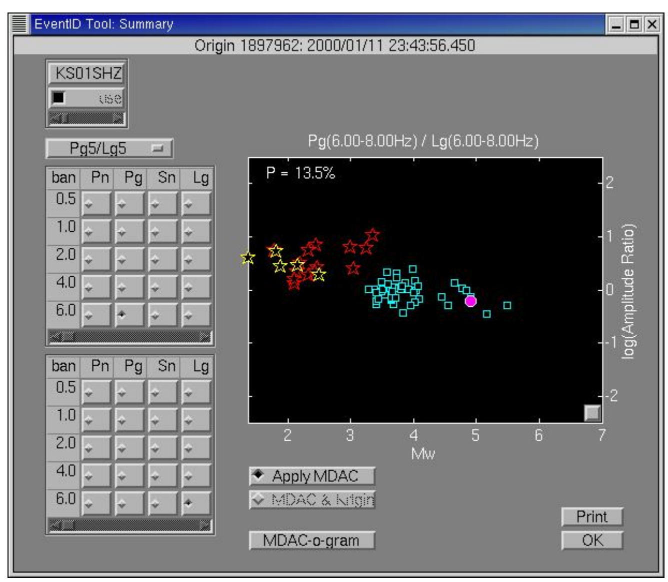

Figure 3. EventID Tool. Reference events are subclassified by event type.

phases and change them if necessary. If they are changed, the measurements will automatically be re-made and the ratios updated in the main display.

New features for Version 1.8 of EventID Tool include the ability to display a single measurement (i.e., not a ratio) and to change the scales on which the ratios/measurements are plotted.

Version 1.9 of EventID Tool will support the use of the origin event type to sub-classify the reference events. This will provide the ability to color code the reference events in order to distinguish between chemical and nuclear explosions.

Dendro Tool 1.8

Dendro Tool performs waveform correlation-based cluster analysis techniques on waveform data. The purpose of Dendro Tool is quite simple: to allow a user to quickly and efficiently determine whether a waveform of interest matches any in the available archives. By arranging the correlations in a hierarchical dendrogram, rather than just determining the most similar waveform, the user gets a much more complete picture of how the current event fits with the archived events. For example, in regions with repeated mining explosions, the mines are often easily identified as distinct clusters, and new mining events can be readily identified as such by association with those clusters.

Dendro Tool 1.8 includes the correlation matrix plot within the main display. This should assist the user in quickly determining the appropriate threshold level for selecting families. Dendro Tool also contains two options to help users attempting to process real-world data. The first option is to ignore channel name and just use the station name when determining which waveforms to cross-correlate (Dendro will still only correlate waveforms from the same station) and which arrivals to include. This option is useful for processing waveform data whose channel may have been renamed over time (i.e., ‘sz’ vs ‘SHZ’) or for associating arrivals that were picked on a channel other than the one the waveforms are on. The second option is to automatically transfer arrivals from the reference station of an array to an individual element. This option is to assist users who may want to use Dendro Tool to process a single element of an array and want to include the arrivals that where made on the reference station.

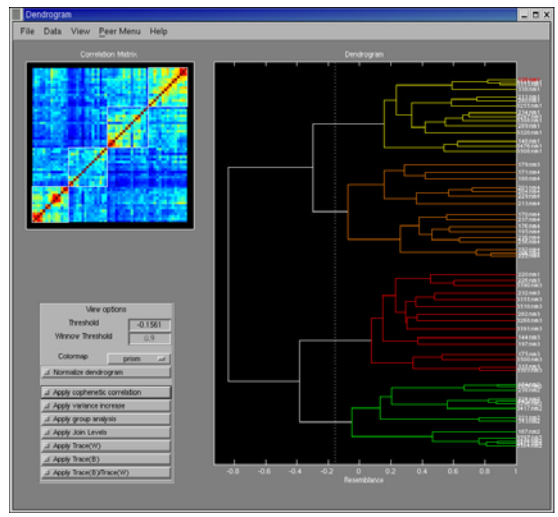

Figure 4. Dendro Tool with the correlation matrix

Infra Tool 1.8

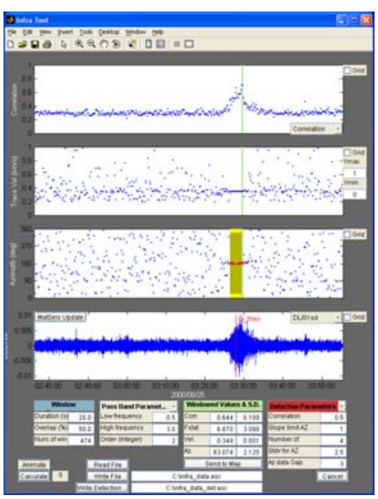

Figure 5. Infra Tool

Infra Tool, included within the MatSeis release, has been expanded to support automatic detection of infrasound signals. Two routines were developed for the task of infrasound signal detection (Whitaker et al., 2002). Both methods are designed to use the results of frequency-slowness signal processing and a set of threshold parameters to determine when signals are detected (Hart, 2004). The detection threshold parameters are of four types: 1) number of point data considered, 2) correlation value to exceed, 3) slope limit bounds with standard deviation, and 4) detection gap points. The final detection parameter that will need to set is Number of Points. This fixes the number of data points used by the detection algorithms in determining the characteristic slope for the azimuth data being analyzed. Once processing is complete, detection results can also be written to a file.

Version 1.9 of Infra Tool will have the ability to add an arrival to MatSeis and set the arrival’s azimuth and azimuth standard deviation based on the results from Infra Tool.

Hydro Tool 1.8

Hydro Tool employs the same array and frequency-slowness processing techniques as Infra Tool, except that the default parameter settings have been changed to values more appropriate for hydroacoustic data. It is also included in the standard MatSeis release. Hydro Tool is intended to provide a platform for future research and development in hydroacoustic processing. Subsequent versions will incorporate any feedback and suggestions that users may have.

Measure Tool 1.8

Measure Tool is the MatSeis utility for performing picking/re-timing arrivals and measuring amplitudes. There are three new features in Measure Tool 1.8: The addition of a spectrogram plot, the ability to set first-motion direction and quality for long-period and short-period signals, and the ability to compute regional event durations using a frequency spectrum analysis technique.

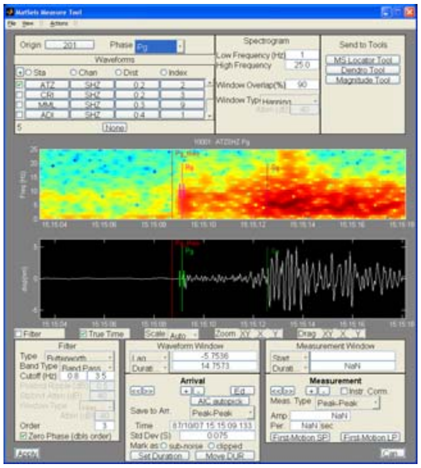

Figure 6. Measure Tool

Tool Communication

The rich suite of KB tools developed by NNSA has provided important new capability for monitoring, but without some sort of inter-tool communication, using the tools for daily monitoring is impractical. In general, each tool has been developed independently without any expectation that it would be able to interface with anything other than a small subset of the KB. Also, the evolutionary method by which the KB has grown has prevented any sort of standardization of practices across the software being developed. The effect of this manner of development has been that we are now faced with a set of tools that have little in common, other than the fact that they are all part of the KB. The tools are written in a variety of programming languages and were each custom developed to address a specific need.

Previous efforts to address this failing have met with mixed results. In the past, changes to the tools have been made to allow specific tools to communicate. This communication has always been point to point between the two tools in question (such as ArcView and MatSeis).

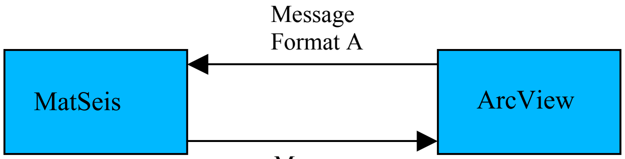

Figure 7. Previous style of tool communication

The messages passed between the tools were in a custom format that was very inflexible, and a new format had to be defined for each communication path. Also, it was necessary to manually implement all of the possible interactions between tools. The end result was a method of communicating that was difficult to create and even more difficult to extend.

The challenges experienced with previous attempts at tool communication have led to a set of requirements that should greatly improve the ability of tools to interact as well as reduce the burden on the individual tool developers. First, the design should allow any tool to communicate with any other tool without having to be modified specifically to implement that interaction. This implies that the set of allowable tool interactions have to be defined at run-time. Second, the design and implementation should require minimal effort on the part of the developer to make use of this communication. The developer should not have to be involved in any of the details relating to how the communication is taking place. Third, the communication should not be platform-specific and a consistent message format should be used for all communication.

Towards this end, we have developed a common library that the tools can make use of in order for them to communicate. By using such a library, all the tools will be guaranteed to be making use of the same compatible communication mechanism. We chose to implement the tool communication in Java and use XML as the message format. This combination of language and message format will provide us with platform independent communication. However, we still have the fact that our set of tools was written in a variety of languages (Java, Matlab, C, C++). Fortunately, we have been able to work around this multiplicity of languages without too much trouble. Java tools can use the communication package directly, and so can the Matlab based tools thanks to Matlab's Java support. C/C++ applications we develop can make use of JNI to interface between the Java and C/C++ code.

The tool communication makes use of the Java Architecture for XML Binding (JAXB). Using JAXB, we are able to write a schema document that is then compiled into Java source code. These JAXB derived Java Objects are able to convert themselves to and from XML, freeing the developers from having to work directly with XML. The use of JAXB has greatly simplified the message handling within the tool communication.

The tool communication also makes use of an NNSA-developed library called DBTools (see Ballard and Lewis, 2004), which is used for intelligently managing database content. DBTools is a very rich library capable of managing any type of relational data that could be stored in a database. By using DBTools as the basis for passing data between applications, the tool communication also has the ability to pass any type of data.

The tool communication consists of two distinct parts. The first part of the communication allows the tools to advertise themselves to each other. The second part of the communication allows each tool to take advantage of the advertisements that it has received to interact with another application.

Peer to Peer Advertisement

In the advertisement portion of the communication, all of the tools connect to a multicast UDP socket. So, all of the KB tools are listening to a single common source. Any messages sent to the Multicast Socket are received by all of the tools.

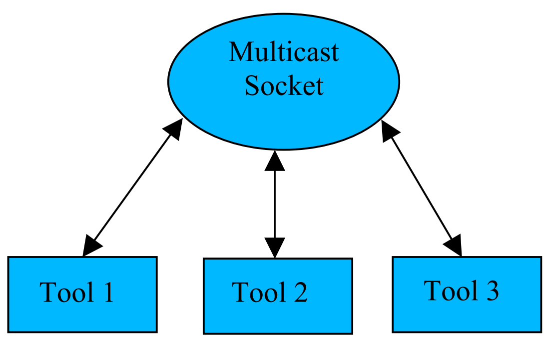

Figure 8. Peer-to-Peer Advertisement using a Multicast Socket

Advertisement messages are sent across the multicast socket. The advertisements contains information such as the name of the tool, the port and address at which the tool is maintaining a server on a TCP socket, and a list of functions that the tool provides. Each function advertisement defines some service that the tool can perform. The advertisement also defines the inputs that the function requires using a DBTools Schema object. The calling application will use the Schema object to determine the type of data to provide as input to the function. An example for MatSeis is shown below:

```
<advertisement>
       <name>MatSeis</name>
       <address>192.168.0.1</name>
       <port>4446</port>
```

```
        <id>2382382832</id>
        <timeout>340983482398</timeout>
        <functions>
                <name>gov.sandia.gnem.matseis.MatSeis</name>
                <method>function1</method>
                <label>Function One</label>
                <description>Perform Function One</description>
                <schema>
                …
                </schema>
        </functions>
        <functions>
                <name>gov.sandia.gnem.matseis.MatSeis</name>
                <method>function2</method>
                <label>Function Two</label>
                <description>Perform Function Two</description>
                <schema>
                …
                </schema>
        </functions>
</advertisement>
```


The advertisements also contain an ID to uniquely identify the tool and a timeout that represents the time when the advertisement expires. Each tool is responsible for periodically sending an updated advertisement to replace the expired advertisement. Each tool also maintains a list of the advertisements that it has received. This caching mechanism eliminates the need to implement any sort of centralized repository (e.g. Tuxedo). Since each tool retains a registry of the tools available and the functions that each provides, it is a trivial process to present the user with a menu detailing these functions.

These messages combine to form a dynamic peer to peer network of tools that are capable of registering themselves with each other. None of the tools need to be coded to specifically interact with any other tool since the introductions are handled at runtime. When a tool exits, it stops sending updated advertisements and so is removed from the registries when the existing advertisements expire.

Though we have not yet tried to implement our communication beyond the applications running on a given user's computer, this should be possible because multicast sockets are not limited to operating on a single host. Since the address of a multicast socket does not correspond to any physical machine address, applications on multiple hosts could all bind to the same socket and would then be included in this network of peer-to-peer tool communication. The messages sent on the multicast socket would propagate through the network as far as the network architecture allowed. This capability holds definite promise for possible future development involving distributed computing. For example, a computationally intensive function could be provided on a high-performance server and that function would be available to everyone on the network. Tools on the users desktop could then access this function as if it were just another application on the desktop.

Function Calls

In the second part of the communication, tools can call the functions that have been advertised. Each tool maintains a TCP socket on which it runs a server. Because the server port number will be communicated at run-time within the advertisement, the next available port on the system can be used. This eliminates the need to track the port numbers for each of the tools in the user environment or some configuration file.

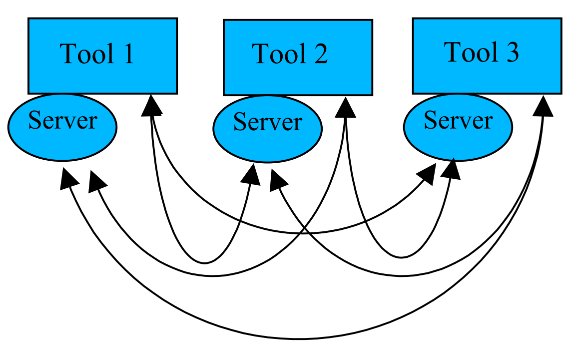

Figure 9. Function Calls between tools.

Using the function advertisement, a tool can assemble a Function Call message to pass to the server. The server then processes the function call by executing the indicated method with the supplied inputs. Once the method has completed, the server collects the outputs from the method and returns them to the calling tool. The inputs and outputs of the function calls are passed as DBTools RowGraph objects.

Although there is some complexity involved in managing the communication for the multicast socket and the server/client code, it is functionality that is common to all of the tool implementations. Therefore, this functionality has been implemented within the communication package and provides a simple interface for each of the tools to make use of. The goal has been to hide as much as possible of the details of the communication and messaging from the individual tool. This should allow the tool communication to develop and expand with minimal impact on the tools.

The Peer to Peer Tool Communication has been incorporated into KB Navigator, MatSeis, and Dendro. In each of these tools, there is a dynamic "Peer Menu" item at the top of the figure. Within the peer menu, there is a submenu for each tool, labeled with the name of the tool. Within each tool submenu is the list of functions that tool has advertised. Using these menus, a user can easily perform any of the functions provided, such as transferring origins between applications. More advanced functions are also available that will allow a user to select origins and stations within KB Navigator and have MatSeis automatically read in the corresponding waveforms.

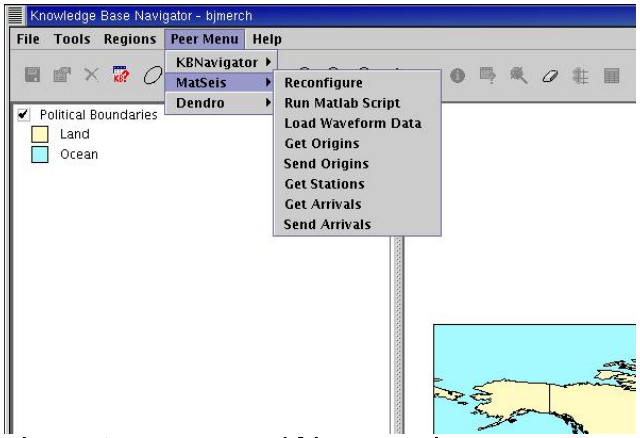

Figure 10. Peer menu within KB Navigator

The utility of the Peer to Peer Tool Communication is expected to increase dramatically as additional tools are incorporated and as the number of functions that each tool can perform increases.

CONCLUSIONS AND RECOMMENDATIONS

In this paper, we have highlighted some of the more significant changes in KB Navigator, MatSeis, and the MatSeis-based regional seismic analysis tools. We have also discussed the tool communication capabilities that are in the process of being incorporated into the various Knowledge Base tools. We expect that the continued development of the individual tools coupled with increasing interoperability between the tools will result in a very powerful and easily extensible regional analysis environment.

The official 1.8 version of the basic MatSeis package for Matlab 6.5 is available to all from the GNEM R&E web site: https://www.nemre.nnsa.doe.gov/cgi-bin/prod/nemre/matseis.cgi

MatSeis 1.9, which will support Matlab 6.5 and 7.0, will be available from the above website in November 2004.

Matlab and the Signal Processing Toolbox are required to run MatSeis. MatSeis will run on Sun workstations, Windows PC's, and Linux PC's. It should run on other supported Matlab platforms as well, but the C code will need to be re-compiled.

ACKNOWLEDGEMENTS

We thank all of the MatSeis users who have helped us to debug and improve the software, particularly our colleagues at LANL and LLNL.

REFERENCES

Ballard, S. and J. Lewis (2004), DBTools: "A Suite of Tools for Manipulating Information in a Relational Database," in Proceedings of the 26th Annual Seismic Research Review Symposium, Orlando, Florida, USA.

Carr, D. (2004), National Nuclear Security Administration Knowledge Base Core Table Schema Document, Sandia National Laboratories, SAND2002-3005 available at https://www.nemre.nnsa.doe.gov/prod/nemre/fileshare/coretables_062104.pdf.

Everitt, B. S. (2001), Cluster Analysis, Arnold, London.

Hart, D. H. (2004), Automated Infrasound Signal Detection Algorithms Implimented in MatSeis – Infra Tool, Sandia Technical Report SAND2004-1889.

Hart, D.H. and Chris J. Young (2002), MatSeis User Manual version 1.7, https://www.nemre.nnsa.doe.gov/cgi-bin/prod/nemre/matseis.cgi.

Hartse, H. E., Taylor, S. R., Phillips, W. S., and G. E. Randall (1997), A Preliminary Study of Regional Seismic Discrimination in Central Asia with Emphasis on Western China, *Bull. Seism. Soc. Am*. 87, 551-568.

Herrin, E. and T. Goforth (1977), Phase-Matched Filters: Application to the Study of Rayleigh Waves, *Bull. Seism. Soc. Am*. 67, 1259-1275.

Mayeda, K. (1993), Mb(LgCoda): a Stable Single Station Estimator of Magnitude, *Bull. Seism. Soc. Am*. 83, 851-861.

Mayeda, K. and W. R. Walter (1996). Moment, Energy, Stress Drop, and Source Spectra of Western United States Earthquakes from Regional Coda Envelopes, *J. Geophys. Res*. 101, 11,195-11,208.

Mayeda, K., R. Hofstetter, A. J. Rodgers and W. R. Walter (1999). Applying Coda Envelope Measurements to Local and Regional Waveforms for Stable Estimates of Magnitude, Source Spectra, and Energy, *21st Annual Seismic Research Symposium on Monitoring a CTBT*, Vol I, 527-533.

Walter, W. R., A. J. Rodgers, A. Sicherman, W. Hanley, K. Mayeda, S. C. Myers and M. Pasyanos (1999). LLNL's Regional Seismic Monitoring Research, *21st Annual Seismic Research Symposium on Monitoring a CTBT*, Vol I, 294-302.

Whitaker, W., Douglas ReVelle, and Tom Sandoval (2002), "On Infrasound Detection and Location Strategies," in Proceedings of the 24th Annual Seismic Research Review Symposium, Ponte Vedra Beach, Florida, USA.