Software Design and Operational Model for the WCEDS Prototype

Judy I. Beiriger, Susan G. Moore, Julian R. Trujillo, and Chris J. Young
Sandia National Laboratories
Sponsored by U.S. Department of Energy
Comprehensive Test Ban Treaty Research and Development Program

ABSTRACT

To explore the potential of waveform correlation for CTBT, the Waveform Correlation Event Detection System (WCEDS) prototype was developed. The WCEDS software design followed the Object Modeling Technique process of analysis, system design, and detailed design and implementation. Several related executable programs are managed through a Graphical User Interface (GUI). The WCEDS prototype operates in an IDC/NDC-compatible environment. It employs a CSS 3.0 database as its primary input/output interface, reading in raw waveforms at the start, and storing origins, events, arrivals, and associations at the finish. Additional output includes correlation results and data for specified testcase origins, and correlation timelines for specified locations. During the software design process, the more general seismic monitoring functionality was extracted from WCEDS-specific requirements and developed into C++ object-oriented libraries. These include the master image, grid, basic seismic, and extended seismic libraries. Existing NDC and commercial libraries were incorporated into the prototype where appropriate, to focus development activities on new capability. The WCEDS-specific application code was built in a separate layer on top of the general seismic libraries. The general seismic libraries developed for the WCEDS prototype can provide a base for other algorithm development projects.

Keywords: waveform correlation, software design, rapid prototype, seismic processing

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED.
OBJECTIVE

The Waveform Correlation Event Detection System (WCEDS) project was initiated in the Spring of 1995 specifically to investigate the potential utility of a waveform correlation-based approach to the CTBT event detection and location problem. While the primary goal of the software design for the project was to develop the WCEDS prototype, an additional goal of the software design effort was to develop a general software base to support rapid prototyping and proof-of-concept testing for CTBT advanced data processing research. To facilitate this, the software design separates the WCEDS-specific software entities and functions from more general seismic software. The two most important requirements for the software system were operation in an IDC/NDC-compatible environment, and incorporating existing IDC/NDC software and commercial third-party libraries to avoid spending development effort for capabilities that could be obtained readily elsewhere.

The WCEDS prototype software development task is primarily a research effort, not a production software development project. Consequently, the software design choices emphasized flexibility, access to intermediate processing results, adaptability to changing requirements, and generality to seismic monitoring. Production software requirements such as execution speed and robustness, though not ignored, were treated as lower priority and developed less fully than for an operational system.

RESEARCH ACCOMPLISHED

The first step of the WCEDS software development effort focused on porting the original code (Shearer 1994) to our system and modifying it to function in an IDC/NDC-compatible environment. The largest portion of this task involved interfacing the code to a CSS 3.0 Oracle database to read waveform data. This initial WCEDS code was used to begin exploring the waveform correlation concept for CTBT application, and provided valuable insight into how to structure a correlation-based prototype detector.

The WCEDS prototype software design began with an analysis of the waveform correlation detector requirements in the context of the general CTBT monitoring problem. The software analysis was based on an object-oriented modeling approach, primarily employing the design process and conventions known as the Object Modeling Technique (OMT) (Rumbaugh, et. al. 1991). The WCEDS analysis model describes what the system is to do, exclusive of considerations of how it might be implemented; it describes the static structure, temporal relationships and behavior, and data transformations required of the system.

After the analysis was completed, we began the WCEDS software system design. This system design describes how the system is structured to meet its requirements, defining the major software components and their organization. We defined the strategy for making design decisions and provided a framework for detailed design and implementation. The system design discusses the components, interfaces, information flow, concurrency, data store implementation, task allocation, resource access, system control, and user interaction. Care was taken to separate WCEDS-specific components and functions from more general seismic monitoring components and functions.

The detailed design and implementation phase is nearing completion. The WCEDS prototype is a fully-developed detection system, reading in waveforms at one end and reporting events and associated arrivals at the other. Other analysis output is possible, such as the waveform correlation and detection threshold comparisons for a specified testcase time and location, a snapshot of the correlation at each grid point at a specified point in time, and a timeline of the correlations at a specified grid point. The detector operation is managed through a Graphical User Interface (GUI); work is continuing to allow the GUI to manage detector parameter configuration and utility program execution as well. A user's manual explains how to use the GUI to run the various application and utility programs.
SYSTEM OVERVIEW

The WCEDS prototype system consists of a set of related executable programs that are accessed and controlled through the WCEDS GUI. This compartmentalized approach allowed the major processing phases of the waveform correlation technique (that is, preprocessing, correlation, and detection) to be developed and investigated independently. It is also an arrangement compatible with the IDC/NDC pipeline scheduling. Operation of the WCEDS prototype is illustrated by tracing a typical data interval through the system from raw waveforms to detected events. The data flow through the system is summarized in Figure 1.

The WCEDS executable programs operate on one data interval and then terminate. They are scheduled and launched with the appropriate input parameters, so that successive data intervals are sequenced through all of the specified processing steps. To initiate a run, the user must first specify the desired configuration and operational parameters to be applied to all of the executable programs during a single WCEDS run. These include:

- the set of phases and time and distance resolution for the travel time models,
- the time and distance resolution for candidate event locations,
- the set of sites to use for seismic data input, and site-specific information for each site, such as channels, site/channel groups, algorithms, filters, recipes, and recipe parameters,
- seismic event and arrival detection thresholds, and optional azimuth and slowness criteria,
- the database interface,
- optional testcase origins,
- the desired start time,
- either continuous operation or a fixed duration,
- which processing steps to perform,
- and preprocessing, correlation, and location interval sizes.

In a typical research scenario, preprocessing, correlation, and event detection are to be performed on a particular block of waveform data. There may be known locations and times of interest. The set of sites includes both arrays and three-component stations, which require different waveform processing algorithms. The block of waveforms is broken into smaller time segments, which are processed consecutively by separate invocations of the appropriate executable program. The executables do not coordinate directly, but are loosely coupled; each starts when its input data becomes available.

First, the waveforms are preprocessed to provide the signal, azimuth, and slowness measurements. The preprocessing interval size specifies the time interval of waveform data for the WCDPreprocessor program, say 15 minutes. The WCDPreprocessor reads the recipes defined for each site and performs the specified processing. Desired processing differences between sites - different filter bands, different algorithms, different input parameters for the same algorithm, etc. - are achieved by defining different recipes. The typical sequence is bandpass filtering of all the waveform channels, a multichannel signal detection method like spatial coherence, scaling of the processed signal, and f-k analysis to take azimuth and slowness measurements at each output data sample time. For each recipe, three processed data streams are stored in the CSS database as waveforms: the processed signal, stored using the recipe name as the channel name, and the azimuth and slowness measurement streams, stored with an ‘A’ or an ‘S’, respectively, appended to the recipe name. The WCDPreprocessor program then terminates. When the next 15-minute interval of raw waveforms is available, another WCDPreprocessor program is started.

Once sufficient preprocessed data becomes available, a WCDCorrelator program is started for the first location interval. The epicentral distance interval from 0 to 180 degrees is discretized to a specified spacing, and the travel time curve for a particular phase is presented as boxcars of finite duration centered at the expected travel time for each discrete distance. The correlation time interval specifies the length of the processed waveforms that are to be correlated against the expected travel time curves, yielding a correlation result for a single postulated origin time. The correlation interval must be long enough to encompass the phases of interest, say 1 hour. The location interval size specifies the time interval of postulated origin times for which trusted correlations will be
FIGURE 1. WCEDS Information Flow.

- User-Specified Operation
  - WCEDS GUI
    - User-Specified Configuration
      - Operational Specification Manager
        - Operational Specification
          - Event Bulletin
          - Testcase Origin Correlation
          - Testcase Timeline

- Transaction Manager
  - Execution Commands

- WCEDS Preprocessor
  - Trusted Data Mask
  - Trusted X Matrix
  - Raw Waveforms
  - Processed Signal
    - Azimuth Measurement
    - Slowness Measurement
  - C Matrix
  - Working X Matrix

- WCEDS Correlator

- WCEDS EventLocator
  - Origins
  - Events
  - Associations
  - Arrivals
made. Trusted correlations are those for which processed data were available for the full correlation time interval. The WCDCorrelator program needs processed data for the location interval plus the correlation interval to compute results for the location interval. It actually produces correlations for postulated origin times as far out as it has data, though results beyond the location interval can not be trusted since full correlation was not possible. The need for the trusted/untrusted model for resolving false correlations near the end of the location interval due to an event in the succeeding interval (see Young, et. al. 1996) became apparent during the initial WCEDS prototype testing.

The desired behavior of the waveform correlation detector is that, for a given postulated origin (grid point and time), the processed signals for each site are correlated with specified phases of the travel time curve at the corresponding site-to-grid point distance and added together to yield a single correlation value for the postulated origin. For performance reasons, the software implementation splits the distance correlation and the summing into two steps. Since any site is expected to be at each of the discrete correlation distances from at least one grid point, and at the same distance from many grid points, its correlation at each discrete distance is computed once and saved. There is also a data mask channel for each processed signal channel that could block segments of the processed waveform from being correlated. Blocked segments could include previously-identified arrivals or data that failed quality control criteria. (Quality control is not implemented in the WCEDS prototype.)

The results are saved in a Correlation Matrix (C Matrix). The C Matrix originally contained, for each origin time, a column for each site and a row for each distance. Each cell contained a single summed value for all of the phase correlations for all of the processed recipe channels for the corresponding site and distance. The C Matrix has since been generalized to hold separate correlation values for individual phases and channels, as well as azimuth and slowness measurements, and to allow different (possibly overlapping) distance ranges for different channels. The azimuth and slowness measurements corresponding to an individual phase correlation are determined as the data samples of the azimuth and slowness channels at the time of maximum correlation between the travel time pulse and the processed signal. In addition, the C Matrix contains a correlation mask (X Matrix). If the phase correlation value does not exceed the specified threshold, the X Matrix is set to prevent it from being included in a sum for some grid point. The user may optionally specify a slowness check for individual phases. If slowness checking is requested, the WCDCorrelator program compares the slowness measurement to the expected travel time slowness for the specified phase. If the slowness values differ by more than the specified threshold, the X Matrix is set to mask that phase correlation value. After the X Matrix is stored, then the WCDCorrelator program terminates. Another will not start until preprocessed data for the next location interval plus correlation interval are available. Correlation intervals after the first will also need phase identification information from the WCDEventLocator, provided as initial X Matrix and data masks for the interval, before they can run.

When a location interval of trusted correlation data (and its corresponding untrusted correlation data) becomes available, a WCDEventLocator program is started. It proceeds by computing, for each postulated origin time in the trusted and untrusted intervals, the correlation sum for each grid point. A C Matrix map defines the summing path for each grid point. The map is a set of offsets from the start of the C Matrix to the individual phase correlations for each site channel at the appropriate site-to-grid point distance. The X Matrix has the same structure, so it may be navigated with the same map. At each map offset, if the X Matrix value allows, the C Matrix phase correlation value is added to the grid point sum. The user may optionally specify an azimuth check for individual phases. If azimuth checking is requested, the WCDEventLocator program compares the azimuth measurement to the site-to-grid point azimuth for the specified phase. If the azimuth values differ by more than the specified threshold, the phase correlation value is not included in the sum.

If the maximum of the correlation sums exceeds the specified threshold, it is considered an event. Associated phases are then identified. Phase identification is accomplished by duplicating the correlation process for the event time and location, but using the phase identification and screening configuration, rather than the detection configuration, of the phase set and the azimuth
and slowness criteria. The screening configuration must at least guarantee that the detected signals will be claimed by this event. Typically, the screening phase set is much larger than the detection set, and the azimuth and slowness checks are much less stringent. A phase correlation that meets the threshold criteria is considered an arrival at the time corresponding to the peak correlation with the travel time boxcar. The WCDEventLocator program stores the event information in the CSS database origin, arrival, and assoc tables. The author, algorithm, and velocity model fields are set to indicate that the record was created by WCEDS. Possible event types are trusted, untrusted, or testcase. Events with times in the trusted interval are also inserted into the event table.

WCEDS tracks the start and stop times corresponding to the travel time boxcar to identify the data segments claimed by arrivals. The data mask for a claimed data segment must be set properly so this arrival will not add into other computations. Similarly, the X Matrix must be set to mask travel time correlations with the data segment. This is done by computing the origin times when the data segment correlated with any phase in the detection travel time model, and masking that phase correlation value. For trusted events, separately-maintained trusted data and X Matrix masks are updated. These trusted masks are used by the WCDCorrelator and WCDEventLocator programs operating on the next location interval. Using the new X Matrix, the WCDEventLocator program then repeats the process of finding the maximum correlation sum. When the maximum sum falls below a specified threshold, or a specified number of events has been found in the interval, the program stores the trusted mask information and terminates. Another WCDEventLocator program will be started when the next location interval of trusted correlation data becomes available.

The user may optionally specify testcase origins or locations for which additional output is produced for analysis. An “event” testcase specifies a target latitude, longitude, and time. The closest grid point and postulated origin time used by the detector are treated as an event, and both the detection and screening correlations are performed. The user may choose to see all phase correlation results, or only those that pass the threshold criteria. A “timeline” testcase specifies a target latitude and longitude. The correlation sum for the closest grid point is output for each postulated origin time, so the seismic activity at that point can be watched in time. A separate utility program, CorrSnapshot, stores the correlation sum for all grid points for a set of origin times and outputs the data so it can be viewed by an animation program.

SYSTEM DESIGN

Architecture

The WCEDS prototype is organized into several subsystems, each providing a particular service to the system through a well-defined and relatively small interface. A subsystem is a package of interrelated software components, associations, and operations that can be designed and implemented independently of other subsystems.

The WCEDS system is structured as a closed architecture of layers and partitions, as shown in Figure 2. The bottom three layers encompass the hardware system and available resources: a distributed network of Sun workstations running the Solaris 2.5.1 operating system, with the Network File System (NFS) and an Oracle database. The top layer provides the WCEDS application - the global capabilities and services - to the user. In between is a layer encompassing the WCEDS prototype functionality, divided into three vertical partitions. The Object Specification Manager Subsystem manages the user specification of the WCEDS components and analysis parameters, providing them to the user interface for display and editing, and providing them to the detector code for operation. The Transaction Manager Subsystem manages the execution of the WCEDS detector according to the user specification; several detector application and utility programs are available. The WCEDS Capabilities Subsystem provides the high-level WCEDS functions and executables that can be launched by the Transaction Manager. It is built upon the Seismic Domain Subsystem, which provides seismological information and processing services. The Seismic
Domain Subsystem is further subdivided into Travel Time Model, Earth Model, Base Seismic Data, Extended Seismic Data, and WCEDS Seismic Domain Subsystems.

FIGURE 2. WCEDS Subsystems Block Diagram

Relationships between the layers are one-way, client-server relationships: a layer knows about the layer immediately below it, but not above it. The WCEDS User Interface Subsystem requests services of the Object Specification Manager Subsystem interactively. It “requests” services of the Transaction Manager Subsystem indirectly, and through it services of the WCEDS Capabilities Subsystem, by specifying parameter files and launching execution. The WCEDS Capabilities Subsystem requests services of the Seismic Domain Subsystem by directly invoking object operations.

Relationships between partitions may be one-way or two-way relationships: a partition knows about the other partitions in the same layer. The relationships between the Object Specification Manager Subsystem and each of the other subsystems in that layer are one-way relationships: the Object Specification Manager provides the operational specification to the other subsystems in a data store. The relationship between the Transaction Manager Subsystem and the WCEDS Capabilities Subsystem is a two-way, peer-to-peer relationship: these subsystems interact through asynchronous, message-based communications. (Currently, this is a one-way interaction: the Transaction Manager launches a WCEDS executable.) The various executables in the WCEDS Capabilities Subsystem, such as the preprocessor, correlator, and locator executables, do not relate to each other directly: they interact indirectly through data maintained in the Seismic Domain Subsystem.

Subsystems

The WCEDS user interaction and object specification management are implemented in the WCEDS GUI executable. The GUI allows the user to view and set the operational control param-
eters for the WCEDS prototype detector: detector start time and duration, which of the preprocessing, correlation, and location steps are to be performed, preprocessing, correlation, and location interval sizes, and whether to start fresh or use event history information. The GUI capability is currently being extended to also manage the WCEDS configuration parameters such as the set of sites, preprocessing recipes, testcase locations, master image configurations, etc. The design allows a user to set configurations, and then store and recall them by name. The WCEDS GUI provides the operational specification to the detector code through a data store. Information for launching WCEDS executable programs is provided to the transaction manager.

The transaction manager is currently implemented as a shell script. It receives user-specified information from the GUI to launch and sequence the executable programs. It is responsible for ensuring that the requested data intervals receive the requested processing, but is not robust. The WCEDS design plans for the transaction manager to be the Distributed Applications Control System (DACS) developed by SAIC for the NDC. We performed some simplified prototyping to verify proof-of-concept, but have not implemented the DACS interface.

![FIGURE 3. Example WCEDS GUI windows.](image-url)
WCEDS capabilities are implemented as separate, but related, executable programs. Coordination and consistency between the programs is provided by the operational specification data store and the scheduling and launch parameters provided by the transaction manager. The major seismic event processing is provided by the programs WCDPreprocessor, WCDCorrelator, WCDEventLocator. There is another variation, WCDCorrLoc, the combines correlation and detection if the interim results are not wanted. In addition, several utility programs for managing the WCEDS prototype have been developed, including programs to build grids and master images, and programs to manage the database.

The travel time model is implemented as a library package called the master image library (libMImage). Travel time information from a known source, such as the IASPEI 91 tables or empirical data collected by UCSD, is interpolated and reformatted to provide expected travel time data at an arbitrary, specified time and distance spacing. The master image uses a specified shape function to spread the expected travel time from a point to a finite-width pulse; for example, the expected time for P to travel 1 degree could be spread from 19.17 seconds to an arbitrarily-shaped pulse with a duration from 14 to 24 seconds. The master image was designed to be a general use library, independent of WCEDS. It has a set of operations that can be called to present the travel time information for requested phases to an application program. The master images used for the WCEDS detector operation to date have been based on the IASPEI 91 travel time curves. At the beginning of the project, we experimented with various time and distance spacings and pulse widths and shapes for the WCEDS application, and eventually settled on a 1-second, 1-degree spacing and a square pulse (boxcar) of variable width chosen so that the phase may span the distance between grid points.

The earth model is implemented as a library package called the grid library (libgrid). It presents geographic location information as a set of discrete grid points (latitude, longitude, and depth) with a specified spacing. It was designed to be a general-use package. It provides geographic functionality such as calculating the distance and azimuth between a station and a grid point and finding the closest grid point to a specified target location, but not WCEDS-specific uses of the grid.

Seismic information is implemented in two library packages, a basic seismic library (libbaseseis) and an extended seismic library (libwcdseis). The base seismic library provides the CSS 3.0 database tables and relations as objects for use by an application program. The complete CSS 3.0 database has not been implemented, but the fundamental tables, such as wfdisc, origin, event, arrival, assoc, site, sitechan, etc. have. The extended seismic library is built on top of the base seismic library, and provides seismic information and functionality beyond the CSS database. In some cases, this is simply adding an attribute or relation to a CSS object. Other common seismic objects like filters, signal processing algorithms, recipes, and station/channel groups are located in this library.

Specific application or implementation uses of the more general seismic domain information is implemented in an application library (libwceds). This includes functions like translating the user specification into operations on the general seismic objects, or customizing them in some way. For example, the WCEDS application allows optional variations of the master image, such as distance-dependent phase scaling or modification of overlapping phases, and uses the grid points as candidate seismic event locations. Libwceds also provides the correlation mappings between the grid points, seismic monitoring stations, and master images, performance-enhancing implementation objects like the C Matrix, non-CSS uses of the Oracle database, and mechanisms for coordinating among the separate WCEDS executables.

Miscellaneous general purpose utility functions are provided in a separate library (libgpu). These include directory path and file name manipulation, string matching functions, file utility functions such stripping comment lines from input files, and date/time utilities.

Where appropriate, the WCEDS prototype incorporates some other libraries not developed at Sandia. A number of NDC libraries are used for seismic functions; examples include waveform filtering, f-k analysis, and input parameter parsing. To minimize the resources spent developing software not fundamental to the waveform correlation or general seismic monitoring problems, commercial libraries were used for a few specific, ancillary purposes. BXPro from Integrated
Computer Solutions allowed a simple GUI to be implemented quickly. DBTools from RogueWave allowed quick development of database access functions. The MATLAB Library and Compiler from The Math Works allowed signal processing algorithms to be developed quickly in the MATLAB environment, and then compiled into the WCEDS executable. The commercial libraries primarily support rapid prototyping; their use in an operational environment would need to be reviewed.

CONCLUSIONS AND RECOMMENDATIONS

The WCEDS software design effort has produced a successful prototype for investigating waveform correlation techniques for CTBT, and has been used in numerous experiments. The WCEDS prototype incorporated existing NDC and commercial software, where appropriate, to concentrate development activities on new capability. Output from the WCEDS prototype includes an event bulletin, origins and associated arrivals inserted into the database, correlation results and data for specified testcase origins, and correlation timelines for specified locations. The WCEDS software development focused on research priorities, such as flexibility and access to intermediate processing results, and not on operational requirements. No concerted effort was made to optimize performance, and it is anticipated that considerable gains could be made in this area. As part of the software development effort, the more general seismic monitoring functionality was separated into a C++ object-oriented library, with the WCEDS-specific application code built on top. In a similar fashion, the general seismic library can provide a base for other algorithm development projects.

REFERENCES


DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.