Waveform server: Web-based display and interaction with high density waveform data recorded by seismic networks as a tool for network diagnostics

Reyes, J.C. <reyes@ucsd.edu>[1], Newman, R.L.[1], Davis, G. A.[1], Vernon, F.L.[1] and Steidl, J.H.[2]

1) Array Network Facility, Scripps Institution of Oceanography, University of California San Diego, La Jolla, CA 92093-0225

2) Institute for Crustal Studies at the University of California Santa Barbara, Santa Barbara, CA 93106

Introduction

The Waveform Server, a Python Twisted application integrated and built on the popular Boulder Real Time Technologies (BRTT) Antelope Environmental Monitoring System allows rapid access and interactivity with multi-station, multi-sensor and multi-channel data stored in Center for Seismic Studies (CSS) 3.0 schema relational databases (Lindquist et al., 2008, Newman et al., 2009, Reyes et al., 2010). The application uses web 2.0 technologies (JSON-based data exchange, AJAX functionality, and HTML5 elements) and standardized libraries (jQuery & jQueryUI) to quickly display large volumes of data in a user-friendly format as either a stand-alone application or remote (iframe) display. The application can essentially be split into two independent copmonents: the server-side Waveform Server Engine, and client side data renderering component (Waveform Viewer).

Waveform Server Uses

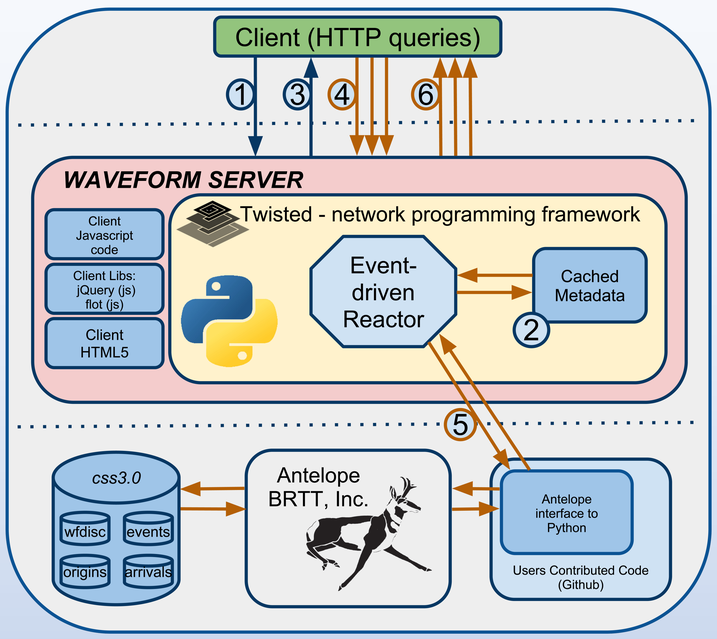
1. Seismic network data return rates and latencies
2. Gap analysis: station specific or network wide
3. Rapid assessment of earthquake waveform data
4. Analysis of state-of-health channel data to assist station engineers in diagnostics

Server-side: The Waveform Server Engine

1. Object-oriented programming (OOP) paradigm: flexible environment for other programmers to expand & improve the application
2. Event driven reactor is a key component of the server
3. Previously used deferred objects (object encapsulates an instruction that will produce a result in the future). This resulted in asynchronous database calls that were colliding and competing for database resources. The latest iteration uses a new multiprocessing system based on a pool of workers maintained by the reactor. The multiprocessing package offers both local and remote concurrency, avoiding a server lock by using subprocesses instead of threads. The multiprocessing module allows the server to fully leverage multiple processors on a given machine with non-blocking independent processing of client requests and data extractions.
4. Data exchange in compressed JSON format is easily parsable by all popular scripting languages. Reformatting data objetcs from tuples to simple arrays has improved server performance.

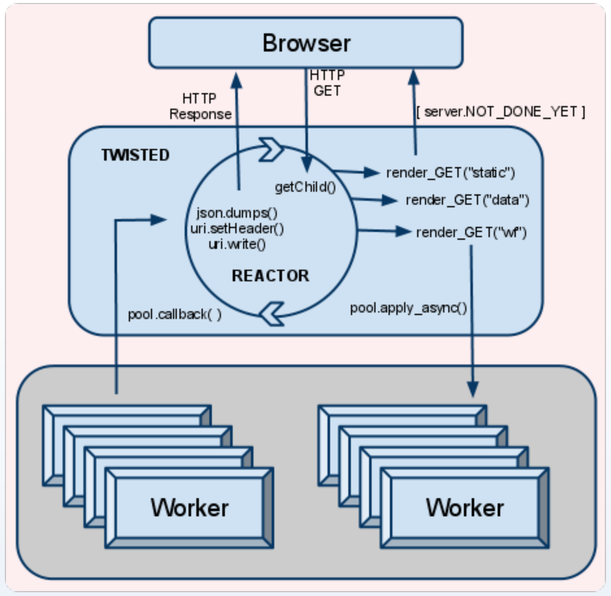
How the engine works: the query lifecycle

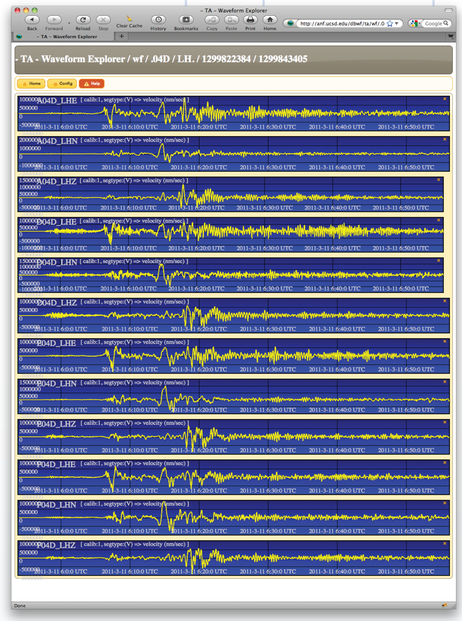
1. Client (i.e. web browser) initiates the process by querying the server with a user provided URI.
2. The Twisted Reactor interprets the URI using cached metadata and produces a meta-query object that contains references to the requested data.
3. This meta-query object gets injected in the HTML application that will return to the browser.
4. JavaScript code on the browser prepares the application (HTML5) through the modification of DOM and CSS elements and initiates independent asynchronous queries to the server to populate the canvas elements with the waveforms.
5. The Reactor will produce Deferred Objects by sequencing the necessary steps to produce the data. Each Deferred Object gets its own thread and the data retrieval process continues out of the Reactor.
6. Each Deferred Object will return independent JSON objects in parallel to the client.



Inside the engine: the reactor loop

1. Client (i.e. web browser) initiates the process with HTTP_GET request.
2. The Reactor loop forwards the request to the apropiate resource.
3. The Reactor acknoleges the query with a response of NOT_DONE_YET
4. The Reactor calls one of the workers from the pool with the function and the request information.
5. The Reactor is ready to accept new connections.
6. Later the worker will finish and the callback function will return the data to the Reactor thread.
7. The Reactor will send the data back using the opened connection to the client.

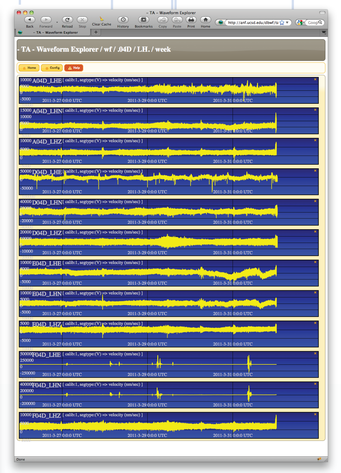




TA records for the Mw 8.9 earthquake in Japan 03/11/11

The USArray project is deploying a series of portable broadband seismic stations across the continental United States over the years 2004 – 2013.

http://anf.ucsd.edu/dbwf/ta/wf/.04D/LH./1299822829/1299826572

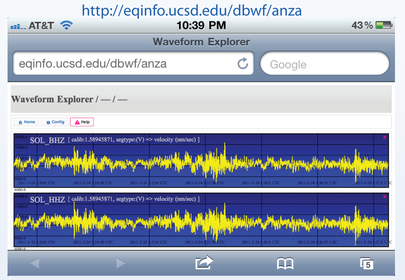


One week of data for four TA stations, LH* channels

http://anf.ucsd.edu/dbwf/ta/wf/.04D/LH./week

(Below) iPhone rendering of waveforms

http://eqinfo.ucsd.edu/dbwf/anza



Client-side: WaveformViewer

Providing anonymous web-based access to both near real-time and archived time-series data recorded by instruments in a seismic network is an easy and efficient way for network operators and scientists to determine network-wide data return rates and quality, scan seismic events, assess calibrations, etc.

The Waveform Viewer interface is easily customizable for any network using simple Antelope configuration files (pf) and templates on the server side, and cascading style-sheets (CSS) for the client-side interface. These can be easily modified to re-architect the client-side interface for any particular purpose.

The application has been thoroughly tested using broadband (1 & 40Hz) seismic data from both the NSF Earthscope Transportable Array (TA) and southern California Anza networks at UC San Diego, and strong motion (200Hz) seismic data from the Network for Earthquake Engineering Simulation (NEES) at the University of California at Santa Barbara. It is also being used by .analysts studing data from the Pinon Flats Observatory in California

Download

You can download the code used in this presentation from the online Git repository hosted by Github.

http://github.com/antelopeusersgroup/antelope_contrib

Future development

1. Real time live interface to streaming waveforms from a Object Ring Buffer (ORB).
2. Optimize performance of data extraction and better handling of long running queries.
3. Promote the development of new clients that can use the server as a gateway to the databases.

References

Reyes, J.C., Newman, R.L., Vernon, F.L., and Steidl, J.H., (2010)Waveform Server: A web-based access to near real-time and archived high-density seismic waveform data, AGU, IN31B-1280

Newman, R.L., Clemesha, A., Lindquist, K.G., Reyes, J.C., Steidl, J.H. and Vernon, F.L., (2009)The Waveform Server: A web-based interactive seismic wavefrom interface, AGU, IN41A-1133

Lindquist, K.G., Clemesha, A., Newman, R.L. & Vernon, F.L., (2008), The Python Interface to Antelope and Applications, Eos Trans. AGU, 89(53), Fall Meet. Suppl., Abstract G43A-0671